

国际信息工程先进技术译丛

CRC Press
Taylor & Francis Group

计算机网络仿真 OPNET实用指南

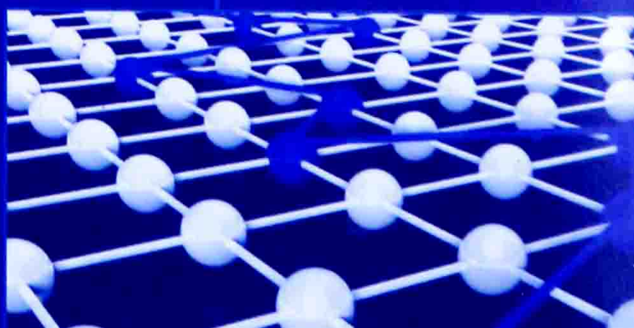
The Practical OPNET User Guide for Computer Network Simulation

(美) Adarshpal S.Sethi
Vasil Y.Hnatyshin 编著

王玲芳 母景琴 等译

机械工业出版社
CHINA MACHINE PRESS





国际信息工程先进技术译丛

计算机网络仿真
OPNET 实用指南

（美）Adarshpal S. Sethi
Vasil Y. Hnatyshin 编著
王玲芳　母景琴　等译





机械工业出版社



图书在版编目（CIP）数据

计算机网络仿真 OPNET 实用指南/（美）塞西（Sethi，A. S），（美）赫纳特新（Hnatyshin，V. Y.）编著；王玲芳等译. —北京：机械工业出版社，2014. 8
（国际信息工程先进技术译丛）
书名原文：The practical OPNET user guide for computer network simulation
ISBN 978-7-111-47060-1

Ⅰ. ①计…　Ⅱ. ①塞…②赫…③王…　Ⅲ. ①计算机网络－计算机仿真－应用软件－指南　Ⅳ. ①TP393. 01－62

中国版本图书馆 CIP 数据核字（2014）第 129917 号

机械工业出版社（北京市百万庄大街 22 号　邮政编码 100037）
策划编辑：张俊红　责任编辑：吕　潇
版式设计：霍永明　责任校对：佟瑞鑫
封面设计：马精明　责任印制：杨　曦
保定市中画美凯印刷有限公司印刷
2014 年 8 月第 1 版第 1 次印刷
169mm×239mm · 27. 75 印张 · 593 千字
0 001—2000 册
标准书号：ISBN 978-7-111-47060-1
定价：99.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务	网络服务
社服务中心：(010)88361066	教材网：http://www.cmpedu.com
销售一部：(010)68326294	机工官网：http://www.cmpbook.com
销售二部：(010)88379649	机工官博：http://weibo.com/cmp1952
读者购书热线：(010)88379203	封面无防伪标均为盗版

本书是市面少有的提供 OPNET IT Guru 和 Modeler 软件全面描述以及如何使用这个软件进行计算机网络仿真和建模的教材，包括 3 部分内容：第 1 部分系统地介绍了 OPNET 的基本功能特征；第 2 部分使用一种自顶向下的方法，描述了如何操作各协议层，每章都解释了有关的 OPNET 功能特征，并包括了一次网络仿真过程中如何使用这些功能特征的各步骤指令；第 3 部分是实验室作业。

本书适合在计算机通信和网络领域工作的 IT 专业人员和研究人员以及该领域的本科生和研究生。





译 者 序

从事计算机网络行业的研究人员、系统部署和维护人员及在校学生，经常面临这样的困境：要进行某种算法或配置的验证，但苦于没有实际的物理网络和物理设备。虽然软性的研究，可以采用机理性的分析或仿真软件进行研究，但由于缺乏可信性的证明，导致在很多情况下仿真结果不为业内人员认可。OPNET 是网络原理性研究和网络实践两方面完美结合的产物，国际上有大学用之进行计算机网络教学的，电信运营商用之验证方案的，研究人员用之研究算法的。在国内普遍用于大学的教学、设备厂商的设备研制，用于研究的还较少。据我们所知，由于该软件系列的学习曲线较长，大部分研究人员和学生都放弃了用之进行科研验证，代之的是一些比较简单的仿真工具。另外还有一个原因，即是缺少 OPNET 比较合适的教材。基于这个原因，在科研过程中，我们尝试寻找一本比较全面的 OPNET 教材，本书正是我们搜寻的结果。通读原书后，感觉有必要介绍给国内网络界的研究人员、从业人员，尤其是大专院校、科研院所的学生。故此，在机械工业出版社的大力支持下，我们翻译了该书。

本书是提供 OPNET IT Guru 和 Modeler 软件全面描述以及如何使用这个软件进行计算机网络仿真和建模的第一本教材。包括 3 部分内容：第 1 部分系统地介绍了 OPNET 的基本功能特征；第 2 部分使用一种自顶向下的方法，描述了如何操作各协议层，每章都解释了有关的 OPNET 功能特征，并包括了一次网络仿真过程中如何使用这些功能特征的各步骤指令；第 3 部分是实验室作业。

第 1 ~ 4 章提供 OPNET 一些基本功能的系统性介绍，仅就任何网络仿真而言也要求提供这样的介绍。这 4 章包括项目和场景的概念，如何创建一个网络拓扑，如何编辑对象的属性并配置拓扑以便反映一个期望的配置，如何配置和运行仿真，以及如何观察和分析仿真结果。第 5 ~ 12 章描述如何使用一种自顶向下的方法操作各协议层。其中第 5 ~ 7 章给出在 OPNET 中如何操作应用，包括标准应用和定制应用，以及如何在网络节点处部署应用和用户概要。第 8 章描述传输层协议：传输控制协议（TCP）和用户数据报协议（UDP）。第 9 ~ 11 章讨论网络层，并包括 IP 寻址的描述和高等 IP 功能的描述。第 12 章描述协议栈的下两层——物理层和链路层，其中包括如何操作以太网和无线 LAN。第 3 部分包含一组实验室作业，每个作业都带有聚焦于每个专题的完整仿真。每个作业都给出了一个完整的仿真，并反映了在主要内容中所述专题的相同进度。各作业描述了对应实验的整体目标，讨论了通用的网络拓扑，并给出完成仿真所需要的系统配置的高层描述。

本书由王玲芳负责第 1 ~ 6 章翻译、全书统稿和校对工作，母景琴负责第 7 ~ 12 章和实验室作业部分的翻译工作。李虹、潘东升、李冬梅、吴秋义、王弟英、吴璟、游庆珍、李传经、王领弟、王建平、李睿、吴昊、王灵芹、张永、李志刚、左会高、申永

林、潘贤才、刘敏、李钰琳、王青改、李倩、陈军、许侠林等同志参加了部分的翻译工作，在此表示感谢。同时感谢机械工业出版社，感谢机械工业出版社的编辑和相关同志。

不过，需要指出的是，本书的内容仅代表作者个人的观点和见解，并不代表译者及其所在单位的观点。另外，由于翻译时间比较仓促，疏漏错误之处在所难免，敬请读者原谅和指正。

译 者

2014年8月于北京

原书前言

对任何IT（信息技术）专业人员而言，拥有各种系统的性能如何仿真和建模的知识成为不可或缺的工具。如今，在没有通过仿真模型对昂贵硬件的性能进行全面评估的情况下，对该硬件的投资是难以有充分理由的。但是，构造一个仿真模型不是一项简单的任务。这要求对仿真和建模概念的透彻理解、被建模系统性质的大量知识以及坚实的数学背景。

OPNET®技术公司（http：//www. opnet. com）是为计算机网络的仿真、建模和分析而开发和销售软件的一家领先公司。IT Guru和Modeler是OPNET提供的用于网络仿真的最流行产品。这些产品被广泛用于教育（教授数据通信和计算机网络中的基本专题和高级专题）以及产业和政府部门（用于各种网络系统的仿真、研究、分析和性能预测）。结果的准确性、丰富的功能以及使用的简单性，是这些软件包吸引了来自各个领域和学科的许多用户的主要原因。

在如今面向技术的世界，除了因特网外，几乎每家公司都依赖于其私有的网络基础设施。因此，对满足如下条件的专业人员的需求在过去数年间显著增长，他们可评估网络性能，并可识别和修复有关的问题。OPNET软件是仿真真实生活的网络、评估它们的性能并在潜在问题出现之前识别这些问题的一项优秀的工具。这是OPNET产品，特别是IT Guru和Modeler，在业界受到广泛应用的原因。它们也被大量用于政府部门，特别是国防部对防御应用和网络的建模。

在各大学中网络课程的教师持之以恒地寻找各种新的活跃的学习工具，吸引学生参与课堂讨论、为学生提供亲自动手的经验，并使学生对专题领域发生更大的兴趣。OPNET软件构成一个卓越的活跃的学习工具，可帮助教师取得所有这些目标。特别地，OPNET软件允许教师依据需要采用简单的或复杂的拓扑，学习和评估多样化的联网系统，形象化地展示各种联网概念，并为学生展示在不同条件下网络性能是如何变化的。毫不令人惊奇的是，全世界有500多所大学[⊖]目前使用OPNET网络仿真软件进行教学和研究。

就我们所知，本书是提供OPNET IT Guru和Modeler软件全面描述以及如何使用这个软件进行计算机网络仿真和建模的第一本教材。本书也包括一组实验室作业，帮助学习该软件的不同方面。

本书的目标读者包括在计算机通信和网络领域工作的IT专业人员和研究人员以及这个领域中课程的本科生和研究生。通过提供OPNET软件丰富功能的深入考察，对于那些正在进行要求产生仿真模型的专业人员和研究人员而言，本书将是一本有价值的参

⊖ http：//www. opnet. com/university_ program/participant_ spotlight/universities. html

考书。同时本书将方便软件的新用户学习OPNET，它以对应于一个网络中协议层的逻辑方式组织专题。实验室作业是以类似方式组织的，并包括到本书主要部分相应专题的参考项。

第1~4章提供OPNET一些基本功能的系统性介绍，仅就任何网络仿真而言也要求提供这样的介绍。这四章包括项目和场景的概念，如何创建一个网络拓扑，如何编辑对象的属性并配置拓扑以便反映一个期望的配置，如何配置和运行仿真，以及如何观察和分析仿真结果。

其他各章描述如何使用一种自顶向下的方法操作各协议层，这种方法是许多网络教材中普遍采用的方法。第5~7章给出在OPNET中如何操作应用，包括标准应用和定制应用，以及如何在网络节点处部署应用和用户概要（profile）。第8章描述传输层协议：传输控制协议（TCP）和用户数据报协议（UDP）。第9~11章讨论网络层，并包括IP编址的描述和高级IP功能的描述，如组播、服务质量（QoS）和IP版本6（IPv6），以及采用路由因特网协议（RIP）和开放最短路径优先（OSPF）协议进行路由。最后，第12章描述协议栈的下两层——物理层和链路层，其中包括如何操作以太网和无线LAN。

每章都包括与该章专题有关的OPNET功能的解释，之后就一个网络仿真过程中如何操作那些功能而提供各步骤的指导。这些功能的逻辑组织，我们打算就其本身而言也作为一个有价值的参考工具。许多这样功能的其他唯一可用的描述是OPNET文档，它是伴随软件产品一起提供的。但是，OPNET文档是极大的，在其中寻找一个特定专题并不容易。本书中各章的组织应该方便了为执行一项期望任务而快速定位指令的工作。

本书的最后部分包含一组实验室作业，每个作业都带有聚焦于每个专题的完整仿真。实验室作业的各专题反映了各专题的相同进度，它是利用这个顺序来组织本书主要章节的。因此，前两项实验室作业本质上是介绍性的，用来形象地说明任何仿真中都需要的主要步骤。在尝试完成这些实验室作业（每项作业都包括针对开发建议阅读的各章的一个列表）之前，建议读者应该阅读本书的第1~4章。实验室作业3~6讨论应用层建模。其他作业沿协议栈向下，从传输层开始，之后到网络层，并以链路层结束。

虽然市场上存在几本其他实验室手册，但没有哪本包括OPNET软件各项功能的详细描述。除了使读者学习如何完成一个网络仿真中各项任务之外，本书也能使读者深入理解OPNET中可用的复杂功能。而且，多数其他的实验室手册为完成每项作业都给出详细的各步骤指令，我们认为这不是一种好的教学手段。在课堂上我们使用这些手册的个人经历中，配置OPNET仿真的各步骤指令是单调的、烦人的和容易出错的。在没有注意到一个步骤时，会非常容易地略过或忽略该步骤，导致仿真的结果不同于实验室手册中报告的那些结果。即使对老练的OPNET用户来说，识别并修正这些错误也是非常困难的。另外，读者还容易在各步骤指令的细节中迷

失，使读者不能注意到任务的总体目标。学生通常盲目地遵循指令，而不尝试通过他们实施的每个步骤进行思考，这使他们不能学习和理解配置步骤。在我们的经历中，学生经常能够成功地完成一项作业，但就他们实际上完成的是什么以及为什么要以那种方式做没有任何提示。

本书中的实验室作业有意没有包含使用 OPNET 软件的各步骤指令。相反，各实验室作业为读者提供试验的总体目标，描述一般的网络拓扑，并给出完成仿真所需要的系统配置的一种高层次描述。实验室作业包含到本书主要部分中各节的引用，其中包含有关如何完成作业中所需每项子任务的各步骤指令。采用这种手段，不知道如何实施任务的一名新读者，都可容易地查找如何完成任务的指令。但随着读者变得越来越有经验，就不太需要为后面的实验室作业查找指令了。我们认为这种方法使实验室作业对读者更有吸引力且更令人激动。更重要的是，读者将更容易透过树木看到森林，并将得到仿真是什么以及为什么要做仿真的更深理解，而不是迷失在细节之中。

应该指出的是，本书仅描述如何操作 IT Guru 和 Modeler 的标准模型。一般而言，Modeler 是一项比较复杂的产品，使用户可设计定制模型，该模型仿真新的协议或现有协议的修改协议。在本书中我们不描述那些功能特征。

最后，我们将感谢本书的评审人员 Jasone Astorga 和 Marina Aguado，感谢他们提出的许多深邃的建议，这些建议有助于极大地改进本书稿的质量。

作者介绍

Adarshpal S. Sethi 博士是在20世纪80年代后期将OPNET软件引入课堂的首批教师之一。自此之后，他将OPNET广泛地用于教学和研究。在纽瓦克的特拉华大学，目前他教授有关计算机网络仿真的一门课程，该课程完全基于使用OPNET进行网络仿真。Sethi博士从印度坎普尔的印度理工学院（IIT）获得计算机科学博士学位，目前他是特拉华大学计算机与信息科学系的一名教授。他曾是位于坎普尔的印度理工学院的教职员工，曾是位于华盛顿普尔曼的华盛顿州立大学的一名访问学者及位于瑞士苏黎世的IBM研究实验室和位于马里兰州阿伯丁的美国军队研究实验室的一名访问科学家。Sethi博士是IEEE网络和服务管理汇刊、网络管理国际期刊和电子商务研究期刊的编委会成员。他也活跃在多个会议的程序委员会。Sethi的研究兴趣包括网络管理、故障管理、无线网络管理的架构和协议，以及网络仿真和建模。2000年，Sethi博士获得特拉华大学的优秀教学奖，该奖项仅授予大学内1200名教学人员中的4名。他也在1996年获得OPNET卓越建模奖，是由Mil3公司（OPNET技术公司以前的名称）授予的，奖励他在一篇研究文章中对OPNET模型的创新使用。2006年6月，在IEEE通讯杂志的封底，OPNET技术公司在教师风采中给出了Sethi的特写。

Vasil Y. Hnatyshin博士在15年来一直在积极地使用OPNET软件产品。他在几个期刊发表他使用OPNET软件的研究工作，并在许多国际会议上做过演讲。Hnatyshin博士的研究兴趣包括因特网中的服务质量（QoS）、传输控制协议/因特网互联协议（TCP/IP）网络、移动自组织网络（MANET）中位置辅助的路由、网络安全、无线通信以及网络仿真和建模。Hnatyshin博士成功地将OPNET软件集成到几门课程中，如数据通信、计算机网络和TCP/IP以及因特网技术，这是他经常在罗恩大学教授的课程。Hnatyshin博士于2003年从位于纽瓦克的特拉华大学获得博士学位，他目前是位于新泽西葛拉斯堡罗的罗恩大学计算机科学系的一名副教授，在那里他使用OPNET软件产品继续他的教学和研究工作。

目　　录

译者序

原书前言

作者介绍

第 1 章　开始使用 OPNET …… 1

1.1　OPNET IT Guru 和 Modeler …… 1
1.1.1　安装 OPNET IT Guru 和 Modeler …… 1
1.1.2　OPNET 许可证服务器 …… 2
1.1.3　安装时创建的文件夹 …… 2
1.1.4　激活可选的产品模块 …… 3
1.2　管理 OPNET 首选项 …… 4
1.2.1　首选项编辑器 …… 4
1.2.2　改变首选项值 …… 5
1.2.3　环境文件 …… 5
1.3　查看文档 …… 6
1.4　操作文件和模型目录 …… 7
1.4.1　文件选择器模式 …… 8
1.4.2　添加模型目录 …… 9
1.5　项目和场景 …… 10
1.6　操作项目 …… 11
1.6.1　打开一个现有项目 …… 11
1.6.2　采用入手引导创建一个新的项目 …… 11
1.6.3　定义一个项目 …… 13
1.7　操作场景 …… 13
1.7.1　创建场景 …… 14
1.7.2　管理场景 …… 14
1.7.3　选择一个场景 …… 16
1.7.4　输入场景组件 …… 16

第 2 章　创建网络拓扑 …… 18

2.1　引言 …… 18

2.2 创建网络拓扑的对象调色板树工具 …… 19
2.2.1 模型命名惯例 …… 20
2.2.2 Internet_ toolbox 调色板中的模型 …… 22
2.3 操作对象调色板树 …… 23
2.3.1 打开对象调色板 …… 23
2.3.2 在对象调色板中搜索模型 …… 23
2.3.3 创建定制模型 …… 24
2.4 创建网络拓扑的方法 …… 25
2.4.1 添加节点 …… 25
2.4.2 添加链路 …… 26
2.4.3 删除节点或链路 …… 26
2.4.4 其他编辑操作 …… 26
2.5 快速配置工具 …… 27
2.5.1 使用 Rapid Configuration 工具创建网络拓扑 …… 27
2.5.2 使用 Rapid Configuration 工具创建以太局域网 …… 27
2.6 配置链路对象 …… 28
2.6.1 改变基本链路性质 …… 28
2.6.2 验证链路连通性 …… 29
2.7 使网络单元失效并恢复 …… 31
2.8 子网 …… 32
2.8.1 添加一个子网对象 …… 32
2.8.2 在网络层次结构中移动对象 …… 34
2.8.3 使用子网创建一个网络拓扑 …… 34
2.8.4 在子网间移动对象 …… 36
2.9 创建拓扑注释 …… 36
2.9.1 将注释调色板中对象添加到项目工作空间 …… 37
2.9.2 修改注释调色板对象的属性 …… 37
2.9.3 在项目工作空间中显示/隐藏注释调色板对象 …… 38
2.10 去除节点杂乱放置 …… 38
第3章 配置网络拓扑 …… 40
3.1 引言 …… 40
3.2 对象属性 …… 40
3.2.1 属性类型 …… 41
3.2.2 对象弹出菜单 …… 41
3.3 编辑属性对话框 …… 43
3.3.1 访问属性描述 …… 44
3.3.2 操作复合和分组属性 …… 45
3.3.3 具有多个实例的属性 …… 45
3.3.4 过滤属性 …… 46

3.3.5　使用常规 Edit Attributes 过滤功能来寻找属性 …… 46
3.3.6　使用高级 Edit Attributes 过滤功能来寻找属性 …… 47
3.4　配置对象性质 …… 47
3.4.1　改变单一对象的属性值 …… 48
3.4.2　改变多个对象的属性值 …… 48
3.4.3　编辑被选中的对象 …… 49
3.4.4　编辑类似节点或链路 …… 49
3.4.5　一个对象的模型属性 …… 49
3.5　提升对象属性 …… 50
3.5.1　提升一个对象属性 …… 52
3.5.2　对一个对象属性实施取消提升操作 …… 53
3.5.3　在仿真层次配置提升的对象属性 …… 53
3.5.4　在仿真层次为提升的属性指定值 …… 54
3.5.5　在子网层次配置提升的属性 …… 55
3.5.6　使用通配符选项将值指派到多个提升的属性 …… 56
第4章　配置和运行一个仿真 …… 59
4.1　OPNET 中的仿真统计 …… 59
4.1.1　统计收集模式 …… 60
4.1.2　确定要收集哪些统计量 …… 61
4.2　选择仿真统计量 …… 62
4.2.1　选择结果窗口 …… 62
4.2.2　为单一特定网络对象选择仿真统计量 …… 64
4.2.3　为整个场景选择仿真统计量 …… 65
4.2.4　选择全局仿真统计量 …… 65
4.2.5　统计信息和数据收集面 …… 65
4.2.6　统计画线风格 …… 66
4.2.7　统计收集模式 …… 67
4.2.8　修改统计收集性质 …… 68
4.3　配置和运行一个仿真 …… 69
4.3.1　Configure/Run DES 窗口：简单模式 …… 70
4.3.2　Configure/Run DES 窗口：详细模式 …… 72
4.3.3　仿真序列编辑器 …… 73
4.3.4　配置和执行单个仿真场景 …… 74
4.3.5　通过管理场景来配置和执行多个仿真场景 …… 74
4.3.6　为提升的属性设置值 …… 75
4.3.7　仿真执行 …… 76
4.4　结果浏览器 …… 78
4.4.1　结果浏览器中的源平板 …… 79
4.4.2　结果浏览器中的结果平板 …… 80

4.4.3 结果浏览器中的预览和呈现平板 …… 81
4.4.4 分析平板 …… 83
4.5 使用结果浏览器查看仿真结果 …… 84
4.5.1 查看当前场景的仿真结果 …… 84
4.5.2 查看网络中一个特定对象的仿真结果 …… 85
4.5.3 查看这个项目和其他项目中各场景的仿真结果 …… 85
4.5.4 比较仿真结果 …… 85
4.5.5 将新的统计量添加到现有图形 …… 86
4.5.6 寻找最大结果 …… 86
4.5.7 采用时间控制器查看结果 …… 87
4.6 管理分析平板 …… 88
4.6.1 隐藏/显示分析平板 …… 89
4.6.2 安排分析平板 …… 89
4.6.3 删除分析平板 …… 90
4.6.4 将平板转换为注释对象 …… 90
4.6.5 以新的结果重新载入分析平板 …… 91
4.7 高级分析平板性质 …… 92
4.7.1 平板区弹出菜单 …… 92
4.7.2 图形区弹出菜单 …… 94
4.8 DES Log …… 96
4.8.1 离散事件仿真选项卡 …… 97
4.8.2 杂项选项卡 …… 98
第5章 标准应用 …… 99
5.1 对 OPNET 中的流量源进行建模 …… 99
5.2 在 OPNET 中流量源模型的类型 …… 99
5.2.1 显式流量模型 …… 99
5.2.2 背景流量模型 …… 101
5.2.3 混合流量模型 …… 102
5.3 在一个仿真模型中包括应用 …… 102
5.3.1 Application Config 设施对象 …… 103
5.3.2 配置标准应用 …… 105
5.4 标准应用的描述 …… 107
5.4.1 数据库 …… 108
5.4.2 电子邮件 …… 110
5.4.3 FTP …… 111
5.4.4 HTTP …… 111
5.4.5 打印 …… 115
5.4.6 远程登录 …… 116
5.4.7 视频会议 …… 116

5.4.8 话音 …… 118
5.5 使用符号节点名字 …… 121
5.5.1 手动配置一个应用的源首选项 …… 122
5.5.2 手动配置一个应用的目的地首选项 …… 123
5.6 应用统计 …… 124
第6章 高级流量生成功能特征 …… 133
6.1 定制应用简介 …… 133
6.2 为定制应用配置任务和阶段 …… 135
6.2.1 Task Config 工具对象 …… 135
6.2.2 指定任务定义 …… 136
6.2.3 指定阶段配置 …… 137
6.2.4 小结：为定制应用配置任务 …… 143
6.3 在 OPNET 中定义定制应用 …… 144
6.4 在 OPNET 中配置定制应用的例子 …… 146
6.5 显式报文产生源 …… 149
6.6 应用需求和流量断续流 …… 152
6.6.1 应用需求 …… 152
6.6.2 流量断续流需求 …… 153
6.6.3 基线负载 …… 156
6.7 定制应用统计项 …… 158
6.8 应用和流量需求的统计量 …… 162
6.9 显式报文产生源和基线负载的统计量 …… 164
第7章 指定用户概要和部署应用 …… 165
7.1 用户概要 …… 165
7.2 指定用户概要 …… 165
7.2.1 Profile Config 工具对象 …… 165
7.2.2 定义一个用户概要 …… 166
7.2.3 一个简单用户概要例子 …… 167
7.2.4 在一个概要内配置应用行为 …… 168
7.2.5 应用行为属性 …… 170
7.2.6 概要行为属性 …… 172
7.2.7 配置用户概要 …… 173
7.3 配置用户概要的例子 …… 174
7.4 使用 Application Deployment Wizard 部署用户概要 …… 178
7.4.1 Network Tree Browser 平板 …… 179
7.4.2 Application Deployment Hints 平板 …… 180
7.4.3 Dialog Box Controls 平板 …… 180
7.4.4 Application Deployment Operations 平板 …… 180

7.4.5 Deploy Applications 选项 …… 181
7.4.6 Edit Destination Preferences 选项 …… 183
7.4.7 Edit LAN Configuration 选项 …… 184
7.4.8 Clear Application Deployment 选项 …… 185
7.5 在不使用 Application Deployment Wizard 条件下部署用户概要 …… 186
7.5.1 配置客户端节点 …… 186
7.5.2 配置服务器节点 …… 187
7.5.3 指定目的地首选项 …… 188
7.5.4 在一个 LAN 对象中指定客户端数量 …… 189
7.5.5 指定一个应用使用的传输协议 …… 189
7.6 在概要配置和应用部署中的常见错误 …… 190
第 8 章 传输层——TCP 和 UDP …… 193
8.1 引言 …… 193
8.2 支持的 TCP 功能 …… 194
8.3 TCP 配置属性 …… 196
8.4 常用传输协议统计量 …… 200
第 9 章 网络层——IP 介绍 …… 204
9.1 引言 …… 204
9.2 基本 IP 配置属性 …… 205
9.2.1 一个端节点模型的基本 IP 配置属性 …… 205
9.2.2 一个核心节点的基本 IP 配置属性 …… 207
9.3 管理 IP 地址 …… 212
9.3.1 IP 地址和掩码 …… 212
9.3.2 识别附接到一条链路的接口的名字 …… 213
9.3.3 IP 地址配置中的常见错误 …… 214
9.3.4 IP 地址的自动指派 …… 215
9.3.5 清除 IP 地址指派 …… 217
9.3.6 以一个指定的 IP 地址标识接口 …… 217
9.3.7 输出 IP 地址分配 …… 218
9.4 配置其他 IP 功能特征 …… 219
9.4.1 IP 压缩 …… 219
9.4.2 路由协议的基本配置 …… 220
9.4.3 配置 IP 接口的不同类型 …… 221
9.4.4 配置 IP 负载均衡 …… 223
9.5 互联网控制消息协议 …… 224
9.5.1 指定 Ping 模式 …… 225
9.5.2 利用 ip_ ping_ traffic 对象部署 IP Ping 需求 …… 226
9.5.3 使用协议菜单部署 IP Ping 需求 …… 227

9.5.4 Ping 统计量 …… 229
9.6 常用 IP 统计量、表和报告 …… 229
9.6.1 IP 统计量 …… 229
9.6.2 可视化和配置报告 …… 233
9.6.3 查看转发表和路由表 …… 233
第 10 章 高级 IP 功能特征 …… 236
10.1 网络地址转换（NAT） …… 236
10.1.1 NAT 概述 …… 236
10.1.2 配置 NAT …… 238
10.1.3 指定地址池 …… 238
10.1.4 指定转换规则 …… 239
10.1.5 在网关接口上部署转换规则 …… 241
10.2 IP 组播 …… 242
10.2.1 在 OPNET 中支持的 IP 组播功能特征 …… 242
10.2.2 部署 IP 组播流量的步骤概述 …… 243
10.2.3 定义组播流量 …… 243
10.2.4 配置源节点 …… 244
10.2.5 配置目的地节点 …… 245
10.2.6 配置路由器节点 …… 247
10.2.7 其他组播配置参数 …… 252
10.2.8 组播统计和报告 …… 252
10.3 IPv6 …… 256
10.3.1 所支持 IPv6 功能特征概述 …… 256
10.3.2 IPv6 编址 …… 256
10.3.3 为 IPv6 网络配置流量 …… 259
10.3.4 其他 IPv6 选项 …… 259
10.3.5 IPv6 统计和其他性能评估选项 …… 261
10.4 服务质量 …… 262
10.4.1 指定全局 QoS 概要 …… 263
10.4.2 指定本地 QoS 概要 …… 271
10.4.3 在一个接口上部署定义好的 QoS 概要 …… 278
10.4.4 结语 …… 280
10.4.5 QoS 相关的统计量 …… 280
第 11 章 网络层路由 …… 282
11.1 引言 …… 282
11.1.1 在一个被仿真网络中部署路由协议 …… 282
11.1.2 配置路由协议属性 …… 285
11.2 采用 RIP 进行路由 …… 286

11.2.1 RIP 简介 …… 286
11.2.2 本地 RIP 配置属性 …… 288
11.2.3 RIP 接口特定的配置属性 …… 289
11.2.4 配置 RIP 启动时间 …… 291
11.2.5 RIP 仿真效率模式 …… 292
11.3 采用 OSPF 路由 …… 293
11.3.1 OSPF 简介 …… 293
11.3.2 OSPF 属性 …… 294
11.3.3 配置 OSPF 进程 …… 295
11.3.4 在路由器接口上指定 OSPF 配置 …… 298
11.3.5 配置 OSPF …… 299
11.3.6 为 OSPF 路由配置链路成本 …… 300
11.3.7 配置 OSPF 定时器 …… 302
11.3.8 配置 OSPF 区 …… 303
11.3.9 配置 OSPF 区边界路由器 …… 304
11.3.10 配置 OSPF 启动时间 …… 305
11.3.11 OSPF 仿真效率模式 …… 305
11.4 通用的路由统计量 …… 306
11.5 查看路由表 …… 309
第 12 章 数据链路和物理层 …… 313
12.1 引言 …… 313
12.2 采用数据链路层技术部署和配置仿真模型 …… 313
12.3 链路模型属性和特征 …… 314
12.4 以太网 …… 316
12.5 令牌环 …… 318
12.6 无线局域网 …… 320
12.6.1 WLAN 配置属性 …… 321
12.6.2 WLAN 统计量 …… 326
12.7 MANET …… 329
12.8 指定节点移动性 …… 332
12.8.1 定义一个节点轨迹 …… 333
12.8.2 配置一个移动性概要 …… 335
12.9 使用无线部署导引 …… 336
实验室作业 1：OPNET 引言 …… 341
L1.1 引言 …… 341
L1.2 创建仿真项目和场景 …… 341
L1.3 创建网络拓扑 …… 341
L1.4 配置网络拓扑 …… 342

L1.5　配置和运行仿真 …… 343
L1.6　详细研究收集的结果 …… 344
L1.7　复制场景，重新运行仿真并比较收集的结果 …… 346

实验室作业 2：简单容量规划 …… 347

L2.1　引言 …… 347
L2.2　对 ABC 公司的网络建模 …… 347
L2.3　评估 ABC 公司的网络 …… 348
L2.4　比较应用性能 …… 349
L2.5　识别最优带宽/成本比率 …… 349

实验室作业 3：标准应用介绍 …… 351

L3.1　引言 …… 351
L3.2　对研究人员公司的网络拓扑进行建模 …… 351
L3.3　在研究人员公司网络中配置和部署应用 …… 353
L3.4　改正第一个配置问题 …… 354
L3.5　改正第二个配置问题 …… 354

实验室作业 4：HTTP 性能 …… 356

L4.1　引言 …… 356
L4.2　创建仿真模型 …… 356
L4.3　HTTP 1.0 和 HTTP 1.1 …… 358
L4.4　带有流水线和不带有流水线的 HTTP …… 359
L4.5　简单的网页 …… 359

实验室作业 5：对定制应用建模 …… 361

L5.1　引言 …… 361
L5.2　创建仿真模型 …… 362
L5.3　应用性能与较短的应用时间 …… 367
L5.4　应用性能及通过多条报文发送应用消息 …… 367

实验室作业 6：最大传输单元对应用性能的影响 …… 369

L6.1　引言 …… 369
L6.2　创建仿真模型 …… 370
L6.3　增加网络中的跳数 …… 371

实验室作业 7：传输协议——TCP 和 UDP …… 373

L7.1　引言 …… 373
L7.2　在没有数据丢失的环境中的 Hermes 应用 …… 374
L7.3　有数据丢失环境中的 Hermes 应用 …… 376

实验室作业 8：TCP 功能 …… 378

L8.1 引言 …… 378
L8.2 Nagle 算法 …… 378
L8.3 端到端时延对 Nagle 算法的影响 …… 380
L8.4 TCP 的 MSS 尺寸对应用性能的影响 …… 380
L8.5 TCP 的接收缓冲尺寸对应用性能的影响 …… 381
L8.6 TCP 拥塞控制 …… 382

实验室作业 9：IP 编址和网络地址转换 …… 385

L9.1 引言 …… 385
L9.2 初步计算 …… 385
L9.3 仿真建立 …… 386
L9.4 配置动态 NAT …… 388
L9.5 配置端口地址转换 …… 389

实验室作业 10：提供服务质量支持 …… 392

L10.1 引言 …… 392
L10.2 设置基线场景 …… 392
L10.3 FIFO 和 RED 比较 …… 394
L10.4 加权的 RED …… 395
L10.5 加权的公平排队 …… 396
L10.6 具有低延迟队列的 WFQ …… 397
L10.7 改变 WFQ 配置 …… 397

实验室作业 11：采用 RIP 进行路由 …… 398

L11.1 引言 …… 398
L11.2 建立初始 RIP 场景 …… 398
L11.3 配置其他 RIP 场景 …… 399

实验室作业 12：采用 OSPF 进行路由 …… 402

L12.1 引言 …… 402
L12.2 建立初始 OSPF 场景 …… 402
L12.3 带有基于报文的负载均衡选项的 OSPF …… 404
L12.4 采用层次结构选项的 OSPF …… 405
L12.5 采用区边界路由器的 OSPF …… 405

实验室作业 13：以太网 …… 407

L13.1 引言 …… 407
L13.2 总线拓扑 …… 407

L13.3　星形拓扑 …… 410
L13.4　集线器和交换机 …… 412

实验室作业 14：无线通信 …… 414

L14.1　引言 …… 414
L14.2　确定通信范围 …… 414
L14.3　通信范围和发送功率 …… 416
L14.4　MANET 通信：DSR 和 AODV …… 416

第1章 开始使用 OPNET

1.1 OPNET IT Guru 和 Modeler

OPNET 技术公司是网络仿真软件的领先开发商以及应用和网络管理问题方面的解决方案提供商，其软件产品被广泛用于尚未成熟（正在出现的）联网技术的研究和开发，用于通信网络、协议和应用的性能评估、测试和调试，并用于许多学院和大学的教学和研究。OPNET 目前提供十几种软件产品和无数专用的模块和模型，这些可容易地为几乎任何如今的网络范型的研究和评估进行定制。

OPNET 软件具有容易使用的图形用户界面，采用简单的拖放动作和数次鼠标单击，可用来构建各种网络配置并测试其性能。OPNET 软件包含大量模型库，可仿真多数现有硬件设备和先进的通信协议。仿真模型的这种丰富性使您可容易地仿真最复杂的计算机网络，并配置实现最新通信技术的协议。OPNET 的 IT Guru 和 Modeler 是最受欢迎的网络仿真软件包中的两款软件。使用各种通信设备、链路、协议和常用的联网技术的固有模型，这两款产品使用户能够研究各种计算机网络。与 IT Guru 不同的是，Modeler 具有附加的功能，使用户可构造新的仿真模型并修改现有模型。

OPNET 一直维持着一项大学计划，为全世界的高质量学院和大学提供免费的软件许可证和打折的技术支持。通过 OPNET 大学计划，基于 IT Guru 商用 9.1 版的 IT Guru 学术版软件包，可被免费下载。

另外，大学计划向符合要求的教员和学生提供 IT Guru 和 Modeler 全功能商用版。为了得到这些全功能的软件包，需要向 OPNET 技术公司提交一项在线申请。一旦 OPNET 大学计划同意了申请，用户将收到一张安装 CD，或将赋予下载安装文件的权限。在申请同意后，OPNET 也提供一天的维护许可证，这使用户可直接联系 OPNET 的技术支持。在存在软件安装问题时，这个许可证对于第一次使用 OPNET 的用户们是有极大帮助的。当前，IT Guru 和 Modeler 支持 Windows 和 Red Hat Linux 操作系统。欲了解更多信息，请访问 OPNET 的大学计划网页资源，在 http://www.opnet.com/university_program 处可找到。

本用户指南依据的是 IT Guru 和 Modeler 软件包的全功能商用版 16.0，这是本书撰写时可用 OPNET 软件最新版。但是，在本用户指南中描述的多数功能也存在于 OPNET 软件的以前各版本中。在本指南中，我们将 IT Guru 和 Modeler 软件简单地称作 OPNET，并仅当有必要在它们之间做出区分时，才使用具体的名字。

1.1.1 安装 OPNET IT Guru 和 Modeler

为了安装 OPNET，用户需要先确认自己的计算机满足指定的系统要求，之后遵循

安装指令，伴随所提供 CD 或网页上的发布文件。一般而言，IT Guru 和 Modeler 安装过程包括如下步骤：

1）安装软件产品。

2）安装 OPNET 模型。

3）安装产品文档。

4）启动和配置许可证服务器。

5）配置 C/C + + 编译器（仅有 Modeler 需要）。

6）配置网页浏览器，阅读软件文档（可被忽略）。

在软件安装阶段，会被提示，指定要在计算机上安装的许可证系统的类型。1. 1. 2 节描述在安装时可被配置的许可证模式。

1. 1. 2　OPNET 许可证服务器

所有 OPNET 产品在执行之前都必须得到一个许可证：OPNET 软件的每个运行实例需要一个许可证。当启动 OPNET 时，要联系许可证服务器。如果存在所请求软件的许可证，那么许可证服务器将分配相应许可证，将开始 OPNET 执行。否则，请求被拒绝，软件将不能运行。典型情况下，在计算机启动时许可证服务器自动启动，并在后台持续运行。

用户可配置 OPNET，支持如下许可证模型之一：

1）Standalone（独立的）——当前计算机将作为本计算机的一个许可证服务器操作；它将仅管理源自这台计算机的许可证请求。如果希望为这台计算机独占使用分配一定数量的许可证，则这样一个选项是有用的。

2）Floating（浮动的）。由这台计算机提供许可证服务——当前计算机将作为一台网络许可证服务器操作；它将管理这台计算机或您所在网络中任何其他计算机产生的许可证请求。在这种情形中，在网络中的所有计算机间共享软件许可证。对于一名系统管理员，建议采用这个选项，他管理网络软件，并将确保无论何时有活跃的 OPNET 用户时，OPNET 许可证服务器在线并在运行。

3）Floating（浮动的）。从远端服务器访问许可证——当前计算机将作为一个客户端，并将联系一台许可证服务器（驻留在您所在网络中的另一台计算机上），在软件启动时得到许可证。当选择这个选项时，将要求您指定 OPNET 许可证服务器的互联网协议（IP）地址和端口号。联系您的网络/应用管理员，验证许可证服务器信息。当在网络中已经存在一台 OPNET 许可证服务器并正在运行时，典型地使用这个选项。

OPNET 也提供一个许可证管理工具，这允许网络/软件管理员启动/停止 OPNET 许可证服务器、添加/取消注册（deregister、消册）软件许可证，并实施其他许可证管理任务。

1. 1. 3　安装时创建的文件夹

在首次安装时，OPNET 在计算机上产生几个新的文件夹。关键文件夹之一包含实

际的软件组件、模型、文档和系统文件。这个文件夹命名为 OPNET，且它通常位于操作系统存储其应用的目录中，如 Windows 中的 C：\ Program Files。另外，OPNET 产生包含用户特定信息的几个文件夹：

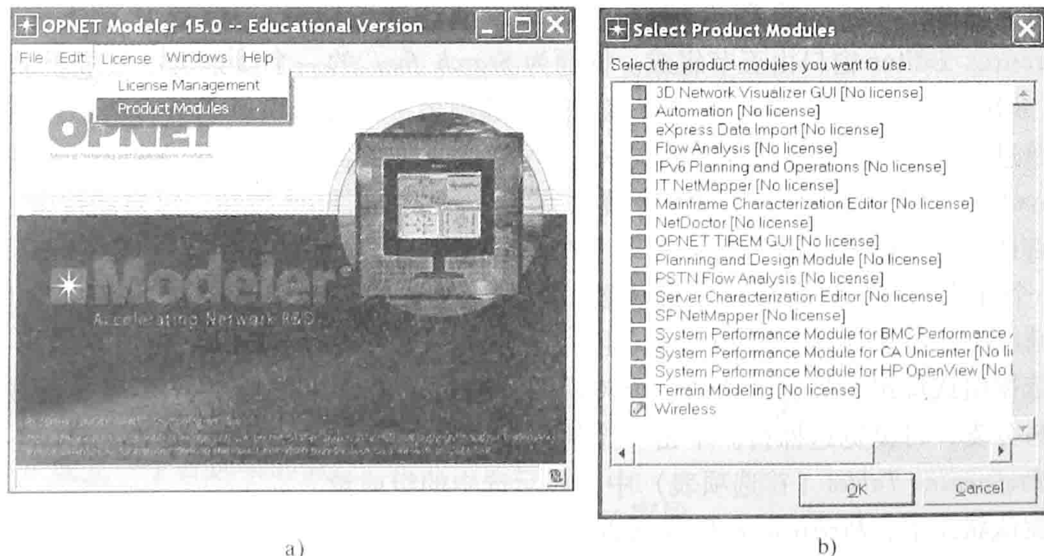
1）op_ admin。这个文件夹包含自动产生的信息，如备份文件（子目录 bk）、临时文件（子目录 tmp）、各种日志文件和一个软件环境配置文件（包含 OPNET 软件首选项）。也将这个文件夹称作 Admin。

2）op_ models。这是 OPNET 存储用户产生的仿真研究配置文件（通常称作项目文件）的默认文件夹。OPNET 也允许将项目文件存储在不同于默认 op_ models 文件夹的位置。

3）op_ reports。这是 OPNET 存储仿真研究报告的默认文件夹。通过修改软件首选项（见 1.2 节），OPNET 支持改变报告的这个默认存储位置。

1.1.4　激活可选的产品模块

在请求时，OPNET 技术公司会为提供附加软件功能的可选产品模块赋予许可证，这些软件功能如地形建模、3D 网络可视化、自动化（automation）等。如果得到这样的许可证，当首次运行 OPNET 时，将显示一个 **Select Product Modules**（选择产品模块）窗口（见图 1.1b）。在首次之后将不显示这个窗口。遵循这些步骤，激活可用的产品模块：



a)　　　　b)

图 1.1　OPNET 产品模块

a）在启动 OPNET Modeler 窗口中的 **Product Modules**（产品模块）选项

b）**Select Product Modules**（选择产品模块）窗口

1）单击可用产品模块的选择框，依据期望选择或取消选择产品模块（见图 1.1b）。

2）单击 OK 按钮关闭窗口。

3）重启 OPNET，从而使改变发生作用。

如果在首次运行 OPNET 时，忘记激活某些模块，则仍然在后来可激活这些模块，遵循如下步骤：

1）启动 OPNET。

2）从 License 菜单（见图1.1a）选择 **Product Modules**（产品模块）。

3）现在出现 **Select Product Modules**（选择产品模块）窗口（见图1.1b）。遵循上面描述的步骤激活期望的模块。

1.2 管理 OPNET 首选项

OPNET 允许改变软件首选项，如系统行为、功能、外观（如色彩和尺寸）、支持的应用、模型目录的位置等。每个首选项由一个变量表示，其值是可被改变的。为了简化搜索一个特定配置选项，首选项被组织成多个分类。一个首选项具有一个“用户友好的”名字和称作标签的一个“技术性”名字。例如，用来阅读 OPNET 文档的程序的首选项，具有用户友好名字 **Path to Document Viewer Program**（文档阅读器程序路径）和技术性名字 **vudoc_ prog**。因为从版本 12.0 开始，产品文档改变为 HTML 格式，所以可将 **vudoc_ prog** 首选项的值设置为 Firefox 或任何其他期望的网页浏览器。

1.2.1 首选项编辑器

可通过图 1.2 所示的 **Preferences Editor**（首选项编辑器），改变 OPNET 首选项。**Preferences Editor** 窗口由五节组成。标题为 *Search for*：的一个搜索文本框位于 **Preferences Editor** 窗口顶部。这个框允许您依据首选项名字、首选项值或所有信息搜索首选项，通过从下拉列表中选择如下值之一：IN Names（在名字中）、In Values（在值中）或 Anywhere（任何地方），您可指定搜索类型。通过单击 **Find** 按钮，可启动搜索。

首选项可依据 *Groups*（组）（即 OPNET 常用配置方面的各分类）或 *Source*（源）（即一个首选项特定集合被定义的位置，如配置文件的一个全路径）。如图 1.2 所示，默认情况下，首选项是依据 *Groups*（组）排列的。**Preferences Editor** 窗口左侧的一节，包含依据组或源组织的首选项的一个树视图。通过单击“+”或“-”号扩展或收缩一个树分支，可浏览这棵树。单击一个树分支，在位于 **Preferences Editor** 窗口右侧的一个 *Preferences Tables*（首选项表）中，显示选中的组或源。

默认状态下，*Preferences Table*（首选项表）仅由两栏组成：*Name*（名字）（它包含一个用户友好的首选项名）和 *Value*（值）（包含那个首选项的实际值）。选择 *Advanced View*（高级视图）检查框，添加两个栏到 *Preferences Table*（首选项表）：*Tag*（标签）（提供首选项的技术名字）和 *Source*（源）（指定在 **Preferences Editor** 外的何处定义首选项）。位于在右侧、*Preferences Table* 下面的 *Preference Information*（首选项信息）面，显示所选中首选项的完全描述。

最后，一组标准按钮位于 **Preference Editor** 底部：

1）**OK**：接收改变并关闭窗口。

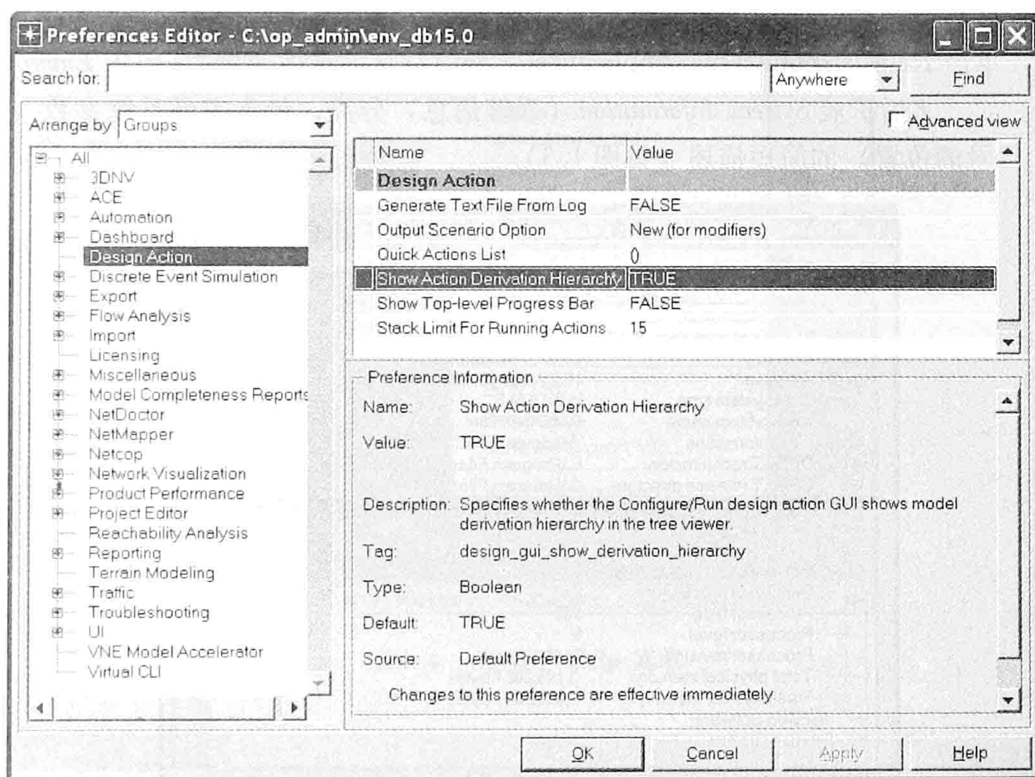


图 1.2　**Preferences Editor**（首选项编辑器）

2）**Apply**（应用）：应用改变，并将窗口留在打开状态。

3）**Cancel**（取消）：在没有保存改变的条件下，关闭窗口。

4）**Help**（帮助）：打开一个帮助窗口，提供 **Preferences Editor** 的详细描述。

1.2.2　改变首选项值

执行如下步骤，查看或改变 OPNET 首选项的值：

1）启动 OPNET。

2）单击 **Edit→Preferences** 菜单项，将打开 **Preferences Editor**。

3）改变一个首选项的值：

① 单击 *Preferences Table* 中期望首选项的 *Value* 字段。

② 从下拉菜单选择一个值。在一些情形中，可能希望选择 Edit 选项，这使您可输入值。

4）单击 **Preferences Editor** 中的 **OK** 按钮，接受所做改变。

1.2.3　环境文件

OPNET 将首选项的值存储在位于 Admin 目录的一个软件环境文件中。为了确定这个目录的位置，遵循下面的步骤：

1）启动 OPNET。

2）单击 **Help→About This Application**（帮助→有关这项应用）。单击 *Environment*（环境）页，之后扩展 *System Information*（系统信息）分类，包含主要系统参数（包括 Admin 目录的位置）的简短描述（见图 1.3）。

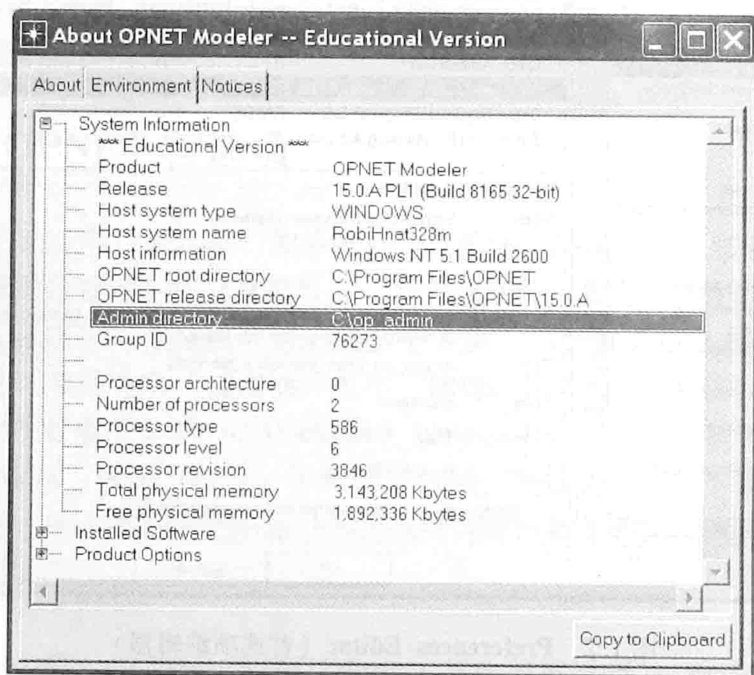


图 1.3 OPNET 的环境设置

Admin 目录包含命名为 env_ db <ver> 的系统环境文件，其中 <ver> 对应于正在使用的 OPNET 版本。例如，OPNET 产品版本 15.0 将具有的环境文件命名为 env_ db15.0。

OPNET 环境文件包含存储于纯文本中的最常用首选项的名字和值。这个文件可采用任何文本编辑器打开和修改。

1.3 查看文档

OPNET 提供详细的产品文档，包括在软件中可用功能的描述，也包括指导（tutorial）和到各种网页资源链接的一个集合。OPNET 文档存储为一个 HTML 文件，并可使用支持 Javascript 和 HTML 框（frame）的网页浏览器直接查看。另一种方法是，通过 **Help** 菜单从主 OPNET 窗口可访问该文档（见图 1.4）。欲了解有关配置网页浏览器和查看在线帮助的更多信息，参见随 OPNET 提供的安装指令。

OPNET 文档组织成主题章和小节。它提供几项有用的功能（见图 1.5），包括搜索、索引、词汇表、快速链接、展开和收缩章和小节标题、显示/隐藏使用软件各种功能的过程描述、访问 PDF 格式的产品文档各章，以及许多其他内容。

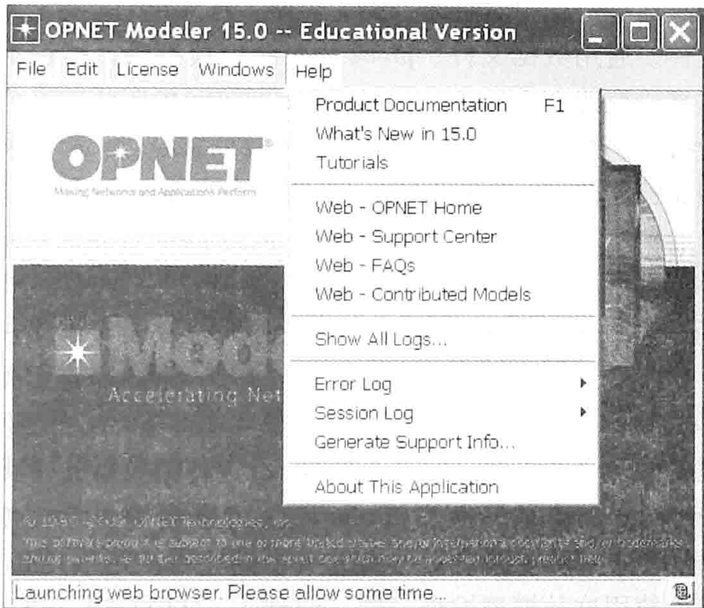


图 1.4　OPNET **Help** 菜单

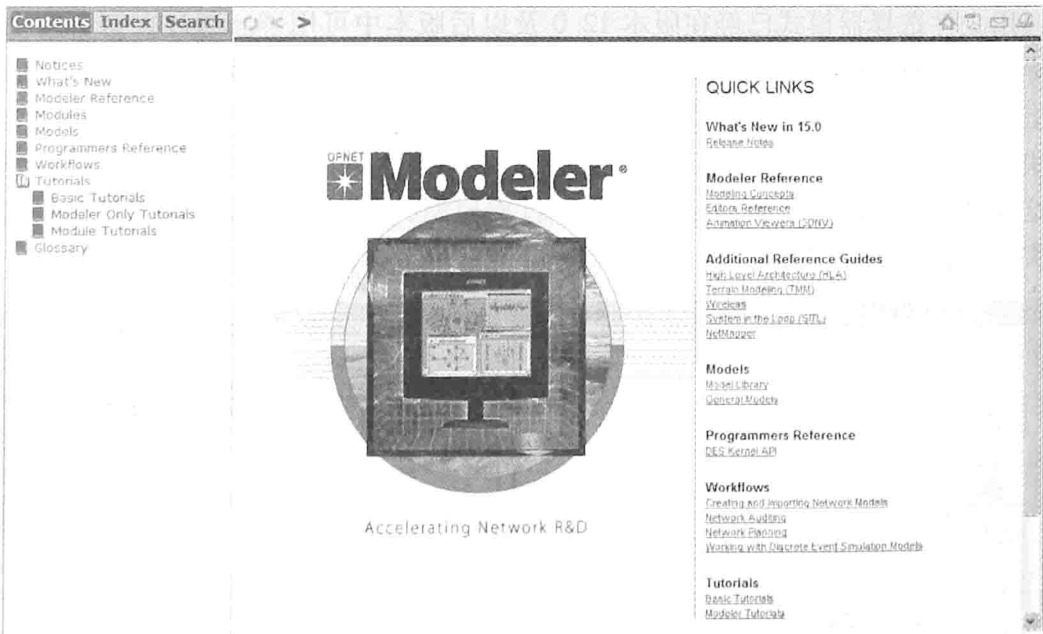


图 1.5　OPNET 产品文档

1.4　操作文件和模型目录

在一次仿真研究过程中的各个点，将需要打开一个现有文件或创建一个新文件并将之存储在一个目录中。IT Guru 和 Modeler 可操作各种不同的文件类型，存储仿真中使

用的网络模型和仿真运行过程中收集的数据。但是，IT Guru 受限于满足如下条件的文件类型，如项目文件、通用数据文件、probe（探针）模型等，即在不修改基础模型条件下，处理创建仿真研究的文件类型，而 Modeler 可访问较广泛的文件类型范围，包括那些包含被仿真设备和网络技术的各实现的那些文件类型。特别地，Modeler 可访问节点模型、过程模型、链路模型、外部 C/C ++ 头文件和实现文件等。

1.4.1 文件选择器模式

OPNET 的最近版本，从版本 12.0 开始，为选择要打开的文件提供两种不同文件浏览方法：

1）*General file chooser*（通用文件选择器）（见图 1.6a）。在当前计算机上支持搜索所有安装的存储设备的一个浏览器。它具有常规 OS 文件浏览器的外观。

2）*File chooser organized by model directories*（依据模型目录组织的文件选择器）（见图 1.6b）。仅显示 OPNET 知道的文件夹的一个浏览器。这些是包括在 OPNET 模型目录列表中文件夹。使用这种模型缩小范围（narrow）可用选项，由此加快选择期望文件的过程。

依据模型目录组织的文件选择器是默认文件选择器，并在 OPNET 所有版本中都存在。通用文件选择器模式已经在版本 12.0 及以后版本中可用了。通过在窗口左下角单击图标，可在模式间切换。在通用文件选择器中，这个图标是一个蓝色方块，其中有一个白星（见图 1.6a）。在依据模型目录组织的文件选择器中，它是分成四个较小的多彩色方块的一个方块（见图 1.6b）。这两种文件选择器模型，使您可依据文件类型浏览文件，这经常会显著地降低定位一个项目、过程模型、C/C ++ 外部代码或其他特定 OPNET 类型文件所需的时间。

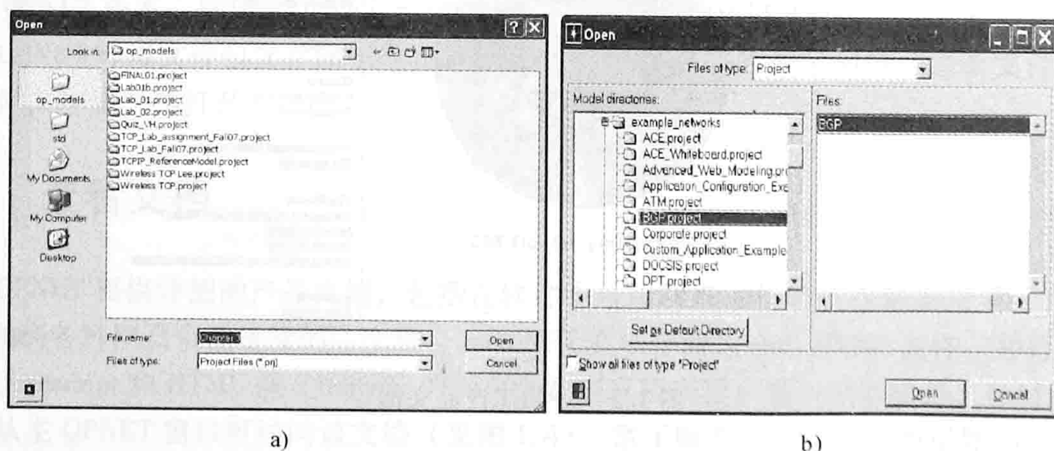


图 1.6 OPNET **File Chooser**（文件选择器）窗口

a）通用文件选择器 b）依据模型目录组织的文件选择器

1.4.2　添加模型目录

OPNET 存储模型文件所用的默认目录是 op_ models（见 1.1.3 节）。经常的情况是，您可能希望将项目文件存储在外部驱动盘或不同于默认模型目录的位置。在这样一种情形中，应该将所关注文件驻留的文件夹添加到 OPNET 模型目录的列表中。这支持对存储于默认目录外面的 OPNET 文件的无缝访问。当采用依据模型目录组织的文件选择器操作时，这个功能是特别有用的，原因是，一旦一个新的文件夹添加到 OPNET 模型目录列表，则位于那个文件夹中的所有文件通过这种文件选择器模式成为可见的。

添加新的模型目录有几种不同方式。下面仅列出一些：

方法 **1**：这是添加一个新模型目录的最简单方法。

1）启动 OPNET。

2）选择 **File→Manage Model Files→Add Model Directory**（文件→管理模型文件→添加模型目录），打开称为 **Browse for Folder**（浏览目录）的一个新窗口。

3）定位要添加的一个新目录，并单击 **OK** 按钮。

4）将出现一个如图 1.7 所示的一个窗口 **Confirm Model Directory**（确认模型目录）。

窗口包括两个检查框：

① *Include all subdirectories*（包括所有子目录）：选择这个检查框，将不仅包括所选中的目录，而且也包括该目录的所有子目录。

② *Make this the default directory*（使这个目录成为默认目录）：选择这个检查框，将使选中的目录成为默认目录，这意味着所有新文件将被保存到那个目录。

5）单击 **OK** 按钮，添加选中的目录。

方法 **2**：这种方法通过 **Preferences Editor**（首选项编辑器），修改名字为 **Model Directories**（模型目录）的软件首选项属性。在 OPNET 的一些较老版本中，这个属性通过其标签名 **mod_ dirs** 引用。

1）启动 OPNET。

2）单击 **Edit→Preferences**（编辑→首选项）。这就打开 **Preferences Editor**（首选项编辑器）（见 1.2.1 节）。

3）搜索属性 **Model Directories**（模型目录）或扩展 **Miscellaneous**（杂项）组[1]，并下滚到 **Model Directories** 属性。

4）在 **Model Directories** 属性的 *Value*（值）字段上单击。一个 **Model Directories** 窗口打开，列出当前包括在这个属性中的所有目录。

5）单击 **Insert** 按钮，并输入要添加到列表的目录的全路径。默认状况下，新近插入的文件夹被放置在列表的顶部，这使之成为 **working**（工作）或 **default directory**（默认目录）。如果在单击 **Insert**（插入）之前，先选中一个目录，那么新目录被插入

[1] 12.0 以前的版本没有将属性归组，因此需要通过窗口滚动条下滚列表，找到 **mod_ dirs** 属性。

在选中目录之前。

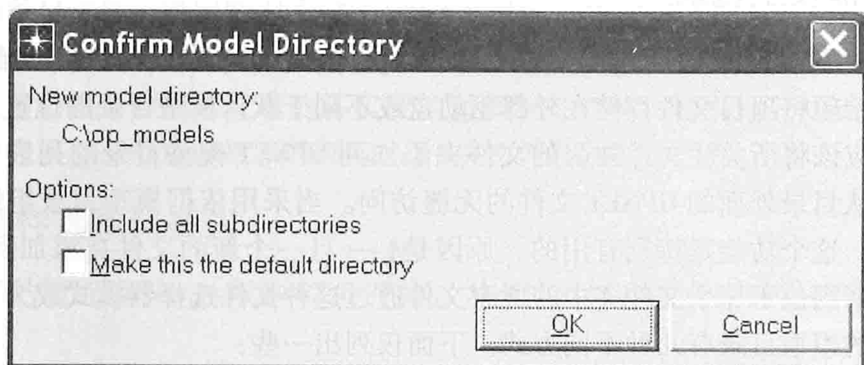


图 1.7 **Confirm Model Directory**（确认模型目录）窗口

6）单击 **Delete**（删除）、**Move Up**（上移）或 **Move Down**（下移）按钮，则可从列表中删除项或操作它们在列表中的顺序。

7）单击 **OK** 按钮两次，接受所做的改变。

添加模型目录的另一种方式是手工编辑位于 **op_ admin** 目录中环境文件内 **mod_ dirs** 属性的值。

在上面的每种方法中，在添加目录之后，好的想法是单击 **File→Manage Model Files→Refresh Model Directories**（文件→管理模型文件→刷新模型目录），这使 OPNET 知道新加目录中的文件。注意，列于 **mod_ dirs** 属性中各目录的顺序是非常重要的，因为它确定 OPNET 软件搜索定位您的文件和模型文件的顺序。例如，考虑如下情况：创建了三个项目，都命名为 MyFirstAssignment，并将它们存储在命名为 c:\ op_ models、c:\ users\ op_ models 和 c:\ users\ project_ files\ op_ models 的三个不同文件夹。假定这些目录以如下顺序列于 mod_ dirs 属性中：c:\ users\ op_ models、c:\ op_ models、c:\ users\ project_ files\ op_ models。

由此，当打开项目 MyFirstAssignment 时，位于目录 c:\ users\ op_ models 中的项目文件将由 OPNET 软件载入，原因是这个目录将在包含 MyFirstAssignment 项目文件的任何其他目录之前被搜索。如果希望打开位于目录 c:\ op_ models 中的项目文件，那么需要将这个目录在 **mod_ dirs** 属性中放在包含项目 MyFirstAssignment 文件的所有其他目录之前。另外，可使用通用文件选择器打开文件。一般而言，好的想法是，将所有模型和目录文件都取不同名字。这样一种策略将保护您免于偶然地重写重要的项目和其他模型文件，或使用一个不正确的模型运行一次仿真。具体而言，可能希望仔细研究系统环境那一章，那里包含了组织用户文件和模型目录的指导方针（guideline）。

1.5 项目和场景

IT Guru 和 Modeler 使用户可在极大量（vast array）仿真环境中实施各种网络技术的研究。OPNET 将各项仿真研究称作项目。每个项目进一步分为一个或多个场景。可

将一个 **project**（项目）定义为，在不同配置范围下，一个特定系统或技术的仿真研究集（collection）。一个 **scenario**（场景）是在一个特定配置设置（即配置参数值的一个集合）下，这样一个系统的单一仿真研究。

考虑如下例子：一家公司将可能实施一项仿真研究，确定如何扩展它的网络基础设施。对于这项研究，公司会创建称为 NetworkExpansion[㊀]的一个项目，项目进一步分为个体场景，称为 10nodes_ 200K、20nodes_ 200K、10nodes_ 500K 和 20nodes_ 500K。一般而言，好的思路是选择描述性的项目和场景名字。在我们的例子中，一个场景名的第一个词指明添加到公司基础设施的节点数，而第二个词指明将新节点连接到公司网络其他部分的链路的容量。

OPNET 产品包含管理项目和场景的各种易用功能，包括如下：创建新的项目和场景、复制现有场景、从一个场景切换到另一个场景、为单一或多个场景配置并运行仿真以及比较来自不同场景和项目的仿真结果。下面讨论一些这样的功能，而其他功能在后面各章描述。

1.6　操作项目

典型情况下，OPNET 中的一项仿真研究，开始时是打开一个现有项目文件或创建一个新的项目文件。

1.6.1　打开一个现有项目

要打开一个现有 OPNET 项目，遵循这些步骤：

1）从主 OPNET 窗口，选择 **File→Open** 或 **Ctrl - O**。

2）选择文件浏览的首选模式（见 1.4.1 节）。

3）确保称为“*Files of type*:”（文件类型）的下拉列表选中“*Project files*”（项目文件）。

4）双击希望打开的项目文件（或单击一次选择项目文件，之后单击 **OK** 按钮）。可能需要浏览可用目录，定位所感兴趣的项目。

1.6.2　采用入手引导创建一个新的项目

1）从 OPNET 主窗口，选择 **File→New**，之后从下拉列表中选择 *Project*（项目），单击 **OK** 按钮。

2）输入期望的 *Project Name*（项目名）和 *Scenario Name*（场景名）（见图 1.8）。

3）确保选中检查框 *Use Startup Wizard when creating new scenarios*（当创建新场景时使用入手引导）。单击 **OK** 按钮。

现在描述 **Startup Wizard**（入手引导）。如果选择了 **Startup Wizard**，那么将出现

㊀ 注意，OPNET 软件的较早版本，不允许项目和场景名中出现空格。

Initial Topology Window（初步拓扑窗口）。否则，将创建一个默认的空项目，之后可开始开发仿真研究。

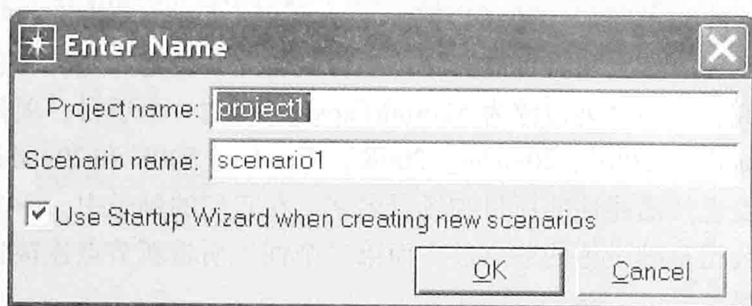


图 1.8　创建一个新项目

1）在 **Initial Topology Window**（初步拓扑窗口），选择 *Create Empty Scenario*（创建空场景），并单击 **Next** 按钮。

2）在 **Choose Network Scale Window**（选择网络规模窗口），在 *World*（世界级）、*Enterprise*（企业级）、*Campus*（校园级）、*Office*（办公室级）、*Logical*（逻辑级）或 *Choose from maps*（从地图中选择）间选择，之后单击 **Next** 按钮。

接下来出现的窗口将取决于您所做的选择。选项 *World*（世界级）、*Enterprise*（企业级）和 *Choose from maps*（从地图中选择）支持通过 **Choose Map**（选择地图）窗口的一个选中地图来定义一个地理规模，而选项 *Enterprise*（企业级）、*Campus*（校园级）和 *Office*（办公室级）支持指定被仿真区域的维度范围，其中通过 **Specify Size**（指定尺寸）窗口使用一个矩形区的宽度和高度。

1）在 **Choose Map**（选择地图）窗口（见图 1.9），在 *Border Map*（边界地图）中选择一项或在可用 *MapInfo* 地图间选择，之后单击“ >> ”按钮，将选中的地图移到选中的区域。一旦选中地理地图，单击 **Next**（下一步）按钮进行到下一个配置选项。

2）在 **Specify Size**（指定尺寸）窗口，指定网络的 *X Span*（X 范围）和 *Y Span*（Y 范围），并单击 **Next** 按钮。

一旦确定了被仿真区域的维度，**Startup Wizard**（入手引导）将打开 **Select Technologies**（选择技术）和 **Review**（回顾）窗口。

1）在 **Select Technologies**（选择技术）窗口，通过单击“*Include?*”下的盒将之从 No 改变为 Yes 来选择模型族，并单击 **Next** 按钮。一个模型族是属于一个技术固有集合（如 *internet_ toolbox*、*ethernet*、*MANET*、*Cisco* 等）的模型集。当选择一个或多个模型族时，在那些族中的对象将成为默认模型族［当 **Object Palette**（对象调色板）打开时出现］的组成部分。不管您的技术选择为何（或甚至您根本没有选择什么技术），在所有技术中的整个对象集（不仅是默认模型族）总是出现在 **Object Palette** 中。

2）在 **Review**（回顾）窗口中，单击 **Finish**（完成）按钮。

3）当使用 **Startup Wizard** 时，可在任何时间单击 **Back**（回退）按钮，则返回到前一窗口。也可单击 **Quit**（退出）按钮，退出 **Startup Wizard** 并创建一个默认项目。

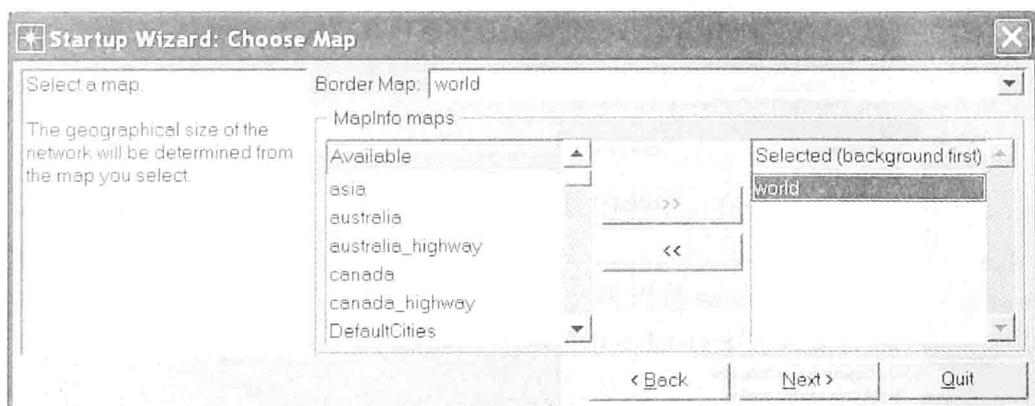


图 1.9　**Choose Map**（选择地图）窗口

当这个过程完成时，将打开 **Object Palette**（对象调色板）和 **Project Editor**（项目编辑器）窗口。现在继续从 **Object Palette** 选择对象，并将它们放到 **Project Editor** 内您的项目工作空间，创建并配置在您的研究中要使用的网络。本指南的后续各章描述如何使用 **Object Palette** 以及如何创建、配置、执行和调试一项仿真研究，之后收集并分析结果。

1.6.3　定义一个项目

经常的情况是，一名新手 OPNET 用户会创建多个项目，其中一些项目可能需要删除。执行如下步骤会永久地删除与一个特定项目关联的所有文件：

1）由主 OPNET 窗口，选择 **File→Delete Projects**（文件→删除项目）。这个动作打开 **Delete Projects** 窗口，它包含所有当前可访问项目的一个列表。

2）为了删除一个特定项目，单击那个项目，之后在出现的 **Confirm**（确认）窗口中单击 **OK** 按钮。这将导致从硬盘上永久地删除所有项目文件。

3）如果需要，重复这个过程，删除任何其他不必要的项目。

4）一旦删除了所有期望的项目，单击 **Close** 按钮，关闭 **Delete Project**（删除项目）窗口。

1.7　操作场景

场景有助于将大型仿真研究组织成较小部分，这些部分详细研究被深入探究现象的特定方面或配置。当希望以一些配置差异［如改变一条链路的数据速率、在一个 LAN 中的工作站数、路由协议、TCP 风格（flavor）、所采用技术等］重复一个仿真时，场景也是有用的。在这些情形中，可在同一项目中创建多个场景，以便更好地组织研究，并方便正在研究的网络各配置的比较。

OPNET 提供了操作项目场景的一系列功能。事实上，在 **Project Editor**（项目编辑器）窗口中可创建和配置仿真研究，它包含称作 **Scenarios**（场景）的一个独立下拉菜

单，专用于管理项目场景的各种选项（见图 1.10）。

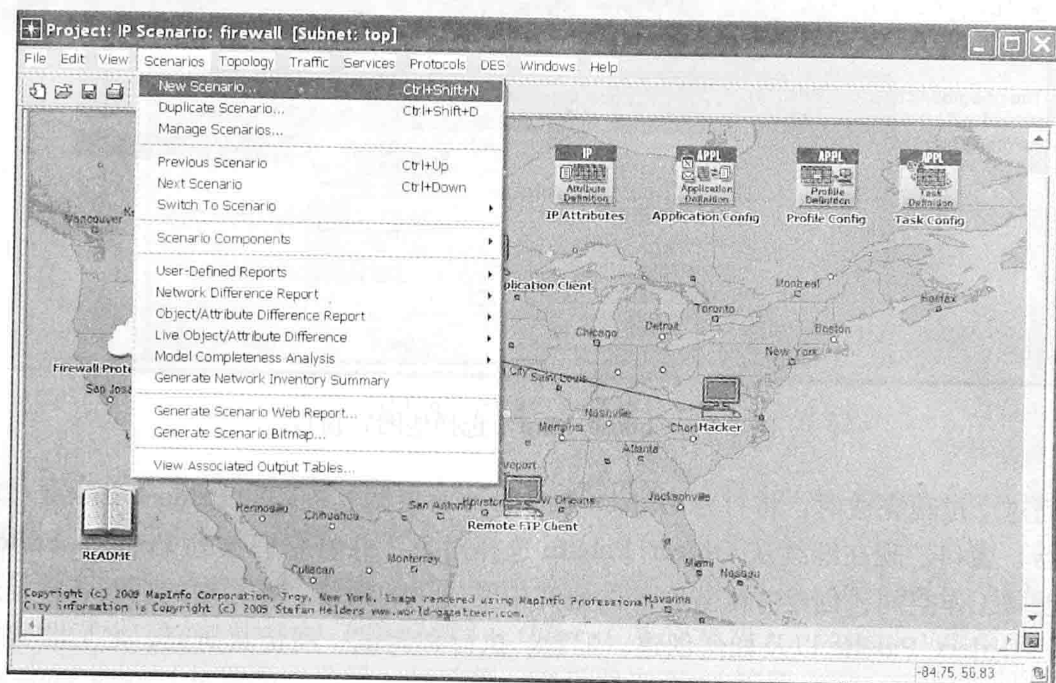


图 1.10 管理项目场景的 **Project Editor**（项目编辑器）选项

下面的各小节描述在 **Scenarios** 菜单上最常用的选项。存在产生、配置、访问和比较各种报告（与当前场景有关，这里不做描述）的附加选项。欲了解有关这些功能的更多信息，参见 OPNET 产品文档的有关发布网络信息章节。

1.7.1 创建场景

Scenarios（场景）菜单中的前两个选项提供创建场景的其他方法：

1）**New Scenario...**（**Ctrl + Shift + N**）在当前项目内创建一个新的空场景。这个选项遵循创建一个新场景的过程，与 1.6.2 节中描述的那样使用 **Startup Wizard** 为一个新项目创建一个空场景相同的过程。

2）**Duplicate Scenario...**（**Ctrl + Shift + D**）复制当前场景。这个选项创建等同于当前一个场景外的另一个场景。当研究一个模型时，这个选项是特别有用的，该模型本质上使用相同的设置，例外是一些小的配置差异。在这样的情形中，需要复制一个现有场景，之后仅修改那些少量不同的配置参数。

1.7.2 管理场景

Scenarios（场景）菜单中的下一选项是 **Manage Scenarios**（管理场景），这是当前项目内管理场景的一个深入全面的工具。当实施大型和复杂仿真研究（由许多场景组成）时，则以单一单击按钮（而不是独立地配置和执行各场景）来配置仿真并开始执行所有感兴趣的场景，是比较实际的。如图 1.11 所示，以如下形式，**Manage Scenari-**

os（管理场景）窗口将所有场景组织在项目内：

1）#指定一个场景号。按下 Ctrl + <场景号>将 **Project Editor** 切换到由输入号指定的场景。单击一个场景的号，为将选中的场景移到表中的一个不同位置，提供了一个选项集合。

2）*Scenario Name*（场景名）指定一个场景的名字。在一个场景的名字上单击，允许改变（即输入）那个场景的名字。

3）*Saved*（存储的）指定当前的场景状态，可以是 saved（存储的）或 unsaved（未存储的）。单击一个场景的 Saved 单元，打开称作 <delete>（删除）的一个选项，当选中时，为删除标记当前场景。

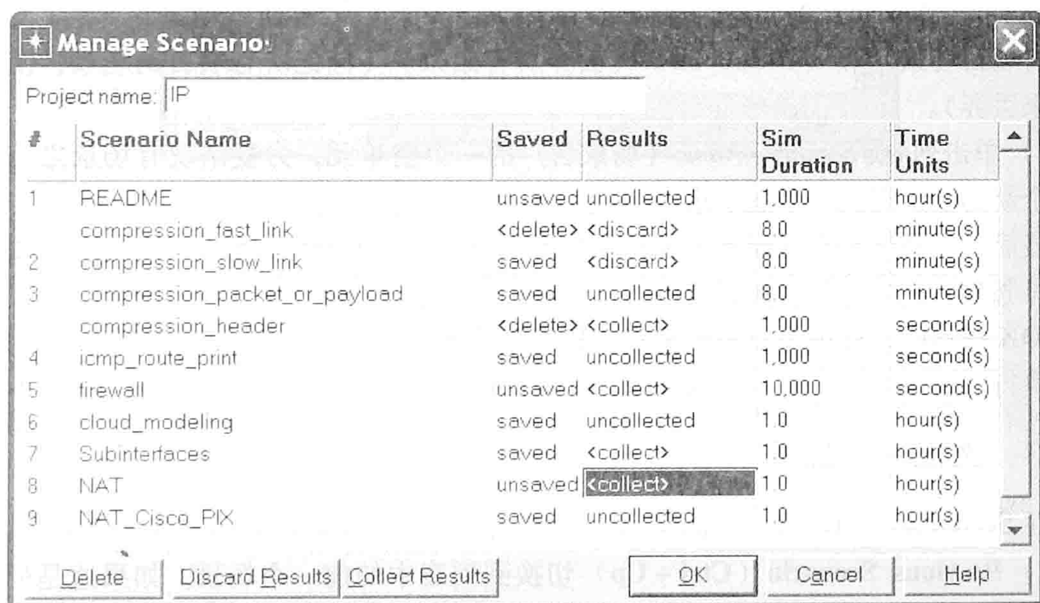


图 1.11　**Manage Scenarios**（管理场景）选项

4）*Results*（结果）指定仿真结果的状态，表示为如下值之一：uncollected（未采集）——为这个场景还没有收集的仿真结果，out of date（陈旧的）——存在这个场景的仿真结果，但由于可能的配置改变变得过时了，或 up to date（最新的）——存在这个场景的仿真结果，且是最新的。在一个场景的 *Results* 单元中单击，提供如下选项：

① <collect>（收集）将配置项目执行当前场景，并重新收集结果。当 *Results* 单元的值设置为 uncollected 或 out of date 时，这个选项才是有用的。

② <recollect>（重新收集）将配置项目执行当前场景，并重新收集结果。当 *Results* 单元值被设置为 up to date 时，这个选项才是有用的。

③ <discard>（丢弃）将配置项目丢弃当前场景的结果。这个选项仅对仿真结果为 up to date 或 out of date 有用。当需要存储磁盘空间时（原因是仿真结果经常要求大量存储空间），这个选项是特别有用的。

5）*Sim Duration*（仿真时长）指定一个仿真场景的执行时长。可单击一个场景的 *Sim Duration* 单元，改变（即输入）仿真的时长（即在单元中输入一个新的值）。

6）*Time Units*（时间单位）指定在相应 *Sim Duration* 单元确定的数值单元。在一个场景的时间单位单元（cell）上单击，使得可从如下值之一选择时间单位：second（s）（秒）、minute（s）（分）、hour（s）（小时）、day（s）（天）或 week（s）（周）。

如果选择场景之一，那么使用按钮 **Delete**、*Discard Results* 或 **Collect Results** 将所选择的场景相应地标记为删除、丢弃其结果或重新收集其结果。

在 **Manage Scenarios**（管理场景）工具中也有几个其他选项：

1）单击列#的标题、*Scenario Name*（场景名称）和 *Saved*，提供选择 <delete all>（删除所有的）（标记要删除的所有项目场景）和 <keep all>（恢复所有的）（取消以前的动作）。

2）单击列 *Results* 的标题，提供选项 <collect all>（收集所有结果）（标志要执行的所有项目场景）和 <discard all>（丢弃所有结果）（标记所有项目的场景，使其仿真结果丢弃）。

3）单击列#或 *Scenario Name*（场景名）的一个空单元，为复制现有场景之一或创建一个新的场景提供一个选项集。

最后，一旦依据喜好配置了项目的场景，那么可单击 **OK** 按钮存储所有改变或单击 **Cancel** 按钮，在不存储改变的条件下，退出 **Manage Scenarios**（管理场景）工具。当单击 **OK** 按钮时，标记为收集或重新收集结果的所有场景，将自动地以其场景号的顺序开始执行。这是一个非常有用的特征，这使您以单次鼠标单击运行多个场景。

1.7.3 选择一个场景

在 **Scenarios** 菜单中菜单选项的下一个集合，处理在当前项目内选择一个场景：

1）**Previous Scenario**（**Ctrl + Up**）切换到列表中的前一个场景。如果这是列表中的第一个场景，这个选项就什么也不做。

2）**Next Scenario**（**Ctrl + Down**）切换到列表中的下一个场景。如果这是列表中的最后一个场景，则这个选项就什么也不做。

3）**Switch to Scenario**（切换到场景）支持切换到当前项目中的一个特定场景。

1.7.4 输入场景组件

Scenarios 菜单中的另一个选项 **Scenario Components**（场景组件），支持输入到或从当前场景各组件输出，包括网络模型、probe（探针）模型（如为收集配置的统计量）、仿真结果和配置信息。默认情况下，输入操作访问位于任何 OPNET - 可见目录中的组件。输入操作覆盖一个场景的现有组件，除非组件自身是一个场景，就不用覆盖了，在这种情形中，场景（们）被添加到当前项目。

为输入一个场景，进行如下步骤（见图 1.12）：

1）选择 **Scenarios→Scenario Components→Import**（场景→场景组件→输入）。

2）从 **Import** 窗口顶部的下拉列表选择组件类型 Project。

3）高亮显示感兴趣的项目。

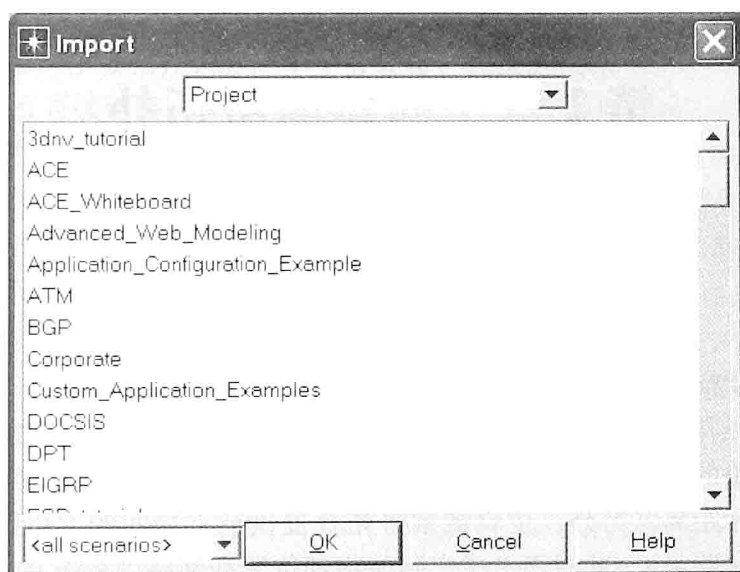


图 1.12　输入类型为 **Project** 的一个场景组件

4）从 **Import** 窗口的左下角的下拉列表，选择感兴趣的场景或选择 <all scenarios>（所有场景），并单击 **OK** 按钮。

在上面的过程中，如果希望输入除 **Project** 外的一个组件类型，那么可从 **Import** 窗口顶部的下拉列表中选择一个不同的值。仅当输入类型为 **Project** 的一个场景组件时，在 **Import** 窗口左下角的下拉菜单才是可用的。

第2章　创建网络拓扑

2.1　引言

网络拓扑用于描述网络设备或设备及其通信信道或链路的互连接，在某个物理空间内的排列。文献经常在一个网络的物理拓扑和逻辑拓扑间做出区分。物理拓扑是节点和连接节点的各链路的实际布局，其中考虑到如下因素，如各节点的物理位置以及跨越连接通信链路的实际距离。另外，逻辑或信号拓扑提供节点之间的通信路径（如数据到达各目的地所取的路由），此时不考虑实际的物理位置和网络中各节点之间的距离。出于为发送和接收数据建立信道的目的，逻辑拓扑主要处理网络设计的协议配置。

本章描述为创建一个仿真网络的物理/逻辑拓扑，在OPNET中可用的特征。一般而言，存在7种主要拓扑（见图2.1）：全连接的网状网（mesh）、非全连接（即部分连接的）网状网（mesh）、bus（总线）、star（星形）、ring（环形）、tree（树形）和hybrid（混合）。在一个全连接网状网拓扑中，每个节点到每个其他节点有一条直接链路。在一个部分连接的网状网中，网络节点具有到一些节点的直接链路，但不必要到所有的其

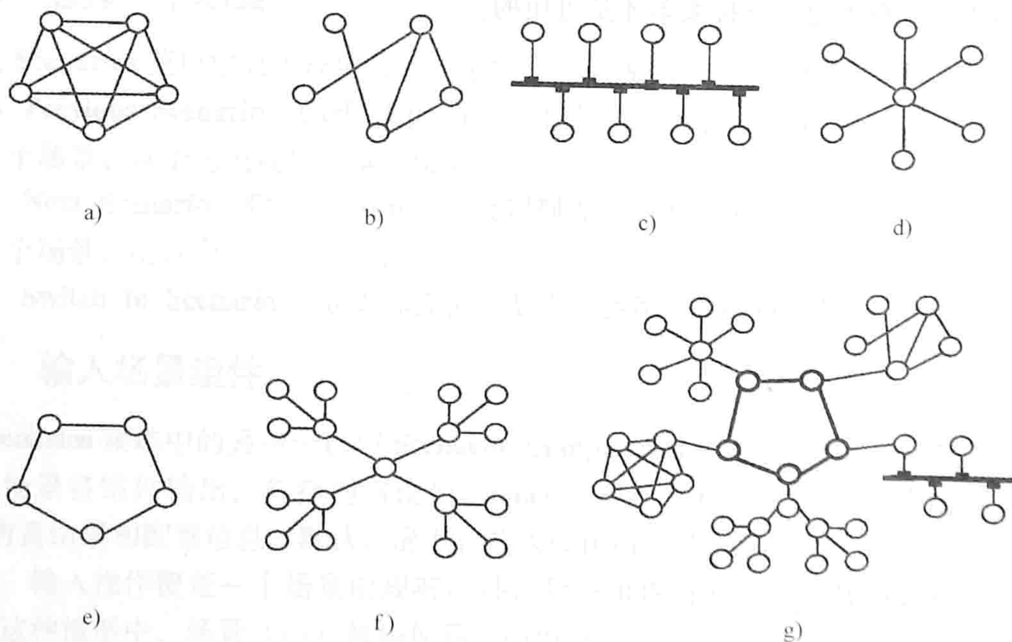


图2.1　网络拓扑

a）全连接的网状网拓扑　b）非全连接的网状网拓扑　c）总线拓扑　d）星形拓扑
e）环形拓扑　f）树形拓扑　g）混合拓扑

他节点。总线拓扑具有单一骨干链路或总线，各节点通过下接（drop）线连接（tapping）到那条链路。星形拓扑具有一个中心设备，连接到网络中的所有其他设备。在非中心节点之间没有直接链路。环形拓扑使各节点以一个环的形式进行组织，其中每个节点在其两侧仅连接到两个节点。在树形拓扑中，任何两个节点通过单一路径连接。最后，混合拓扑使用上述拓扑中的任何拓扑的某个组合，使节点相互连接。

可在具有多条链路（或接口）的节点和具有单一链路的节点之间做出区分。集线器、交换机、路由器/网关和其他类似网络设备负责连接其他节点，由此具有多条链路。端节点（如工作站和服务器）通常通过单一链路连接到其他设备或网络，虽然有时端点会是多穴连接的（即连接到一个以上的网络），且由此会有多条链路。例如，为提高可靠性和/或提供对链路失效的抑制能力，一家公司可具有多条冗余链路，将他们的服务器连接到多个网络。

OPNET 为网络设备、通信链路、LAN（局域网）和网络云提供一个大型的模型库。各模型基于其类型（如链路、集线器、交换机、路由器和服务器）做出区分，并支持通信接口。可用的模型例子包括 Ethernet server（以太网服务器）、点到点协议（PPP）工作站、带有以太网和 PPP 接口的 IP 路由器、100BASE - T LAN、以太网/光纤分配数据接口（FDDI）交换机、100BASE - T 以太网链路、56kbit/s PPP 链路及其许多其他模型。本章描述创建和编辑网络拓扑在 OPNET 中可用的各种功能。具体而言，在本章后面部分接下来的各节，使用 **Object Palette**（对象调色板）和 **Rapid Configuration**（快速配置）工具，创建网络拓扑的指令，描述常用的节点和链路模型，给出如何操作子网，并给出标注工具的概述。

2.2　创建网络拓扑的对象调色板树工具

Object Palette 工具提供到所有 OPNET 模型的访问。通过这个工具，可创建期望的网络拓扑、组织频繁使用的模型（方便容易访问），并创建定制的设备模型。图 2.2 给出了 **Object Palette Tree** 的一个视图，默认对象调色板设置为 *internet_ toolbox*。当使用 **Startup Wizard**（见 1.6.2 节）创建一个新的项目时，自动打开 **Object Palette Tree** 窗口。通过在工具栏中单击 **Open Object Palette**（打开对象调色板）图标，或从下拉菜单中选择 **Topology→Object Palette**（拓扑→对象调色板）选项，可在 **Project Editor** 内打开 **Object Palette**。

可浏览 **Object Palette Tree**（对象调色板树）或使用搜索设施确定仿真研究所需的节点和链路模型。一旦找到一个模型，则可拉并放到 **Project Editor**（项目编辑器）工作空间（或简单的项目工作空间）。**Object Palette Tree**（对象调色板树）将可用模型组织成几个调色板或分类：

1）*Node Models*（节点模型）调色板包含通信设备的可用节点，如集线器、路由器、网关、工作站和服务器。

2）*Link Models*（链路模型）调色板包括链路模型，如 1000BASE - T 以太网链路、

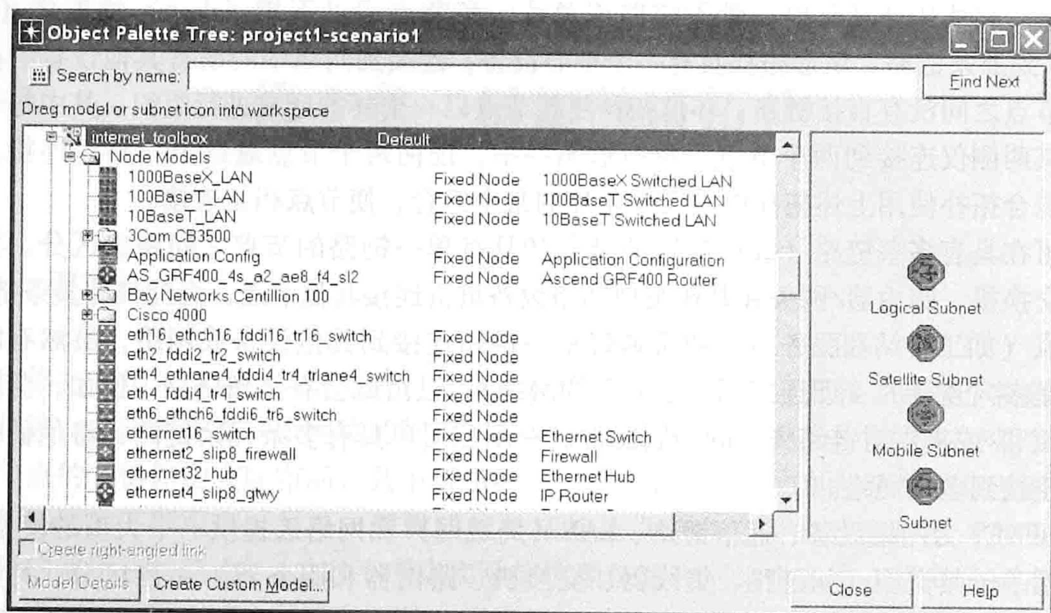


图 2.2 **Object Palette Tree**（对象调色板树）的默认视图

T1 双工链路和 16Mbit/s 令牌环。

3）*Path Models*（路径模型）调色板包含指定网络路径所用的模型，这些路径支持这样的技术，如高可靠因特网协议加密器（HAIPE）、多协议标签交换（MPLS）和公众交换电话网（PSTN）。

4）*Demand Models*（需求模型）调色板包含指定流量流和连接（如 IP 话音流量流、PSTN 话音流量流和 IP 安全）的模型。

5）*Wireless Domain Models*（无线域模型）调色板包含代表无线域（如稀疏网格、全网格和移动性）的模型。

6）*Shared Objects*（共享对象）调色板包括依据共同性质分组的节点、链路、路径、需求和域模型的一个集合。例如，*3Com* 集合包含由 3Com 公司生产的设备的模型。*applications*（应用）调色板包含指定和部署应用所需的模型，而 *internet_ toolbox* 组则包含常用于对互联网建模的节点、链路和工具（utility）模型。

2.2.1 模型命名惯例

在 **Object Palette Tree**（对象调色板树）中每类模型都进一步依据名字、机器类型、对象 ID、厂商、接口类型、链路传输模型和其他参数组织成子群组。这些类经常会重叠，这意味着同一模型可由多个调色板检索。每个模型名通常包括如下信息，如节点或链路类型（如服务器、网关和 LAN）、可用接口（如异步传递模式［ATM］、串行线路互联网协议［SLIP］、FDDI、令牌环和以太网）以及每类接口的数量。

图 2.3 给出在项目创建过程中在 *internet_ toolbox* 调色板中存在的模型的部分列表，默认调色板或由 **Startup Wizard** 选中的模型族。仔细研究名为 *CS_ 4000－3s_ e6_ fr2_ sl2_ tr2* 的高亮显示的节点模型，方法是解析其名字。名字的第一部分指明这是 CIS-

CO C4000 路由器的一个模型，它是一个固定节点（即不是一个移动节点或卫星节点）。这台设备的模型名字解析如下：



图 2.3　*internet_ toolbox* 子类中的一些模型

1）*CS*——Cisco（3C 表示 3Com，AS 表示 Ascend 等）。

2）*4000*——Cisco 路由器的模型。

3）*3s*——三个槽。

4）*e6*——六个以太网接口。

5）*fr2*——两个帧中继（FR）接口。

6）*sl2*——两个 SLIP 或简单的 IP 接口。

7）*tr2*——两个令牌环接口。

类似地，*eth6_ ethch6_ fddi6_ tr6_ switch* 是有 6 个以太网、6 个 EtherChannel（以太信道）、6 个 FDDI 和 6 个令牌环接口的一台交换机；*ethernet4_ slip8_ gtwy* 是具有 4 个以太网接口和 8 个 IP 接口的一台 IP 网关路由器；*ethernet_ wkstn* 是具有单个以太网接口的一个客户节点；*ppp_ server* 是具有支持单个 SLIP 接口的一条点到点链路的一台服

务器。

链路模型的命名惯例非常类似于节点的那些命名惯例。模型名为确定所支持的通信协议和链路容量提供了足够的信息，而模型描述指明链路提供双工通信还是单工通信。例如，如图2.3所示，*1000BaseX* 是双工以太网1000BASE－X链路的一个模型；*FR_ T1* 是具有T1（或DS1）容量的一条双工FR链路；*PPP_ 28K* 是支持SLIP连接的点到点双工28 kbit/s链路的一个模型。

总线拓扑要求稍微不同的链路模型。具体而言，为了创建一个总线拓扑，需要总线和总线分支（bus tap）模型，它们分别存在于 **Object Palette Tree** 中 *Link Models... Bus Models* 和 *Link Models... Bus Tap Models* 分类之下。

OPNET也提供完全网络的一些模型。例如，*10BaseT_ LAN* 对节点通过10BASE－T以太网链路连接的一个LAN进行建模，*atm32_ cloud* 表示有32个ATM接口的一个ATM网络，而 *ip32_ cloud* 是有32个SLIP接口的一个IP网络。*ip32_ cloud* 模型普遍被用来仿真互联网。最后，工具（utility）节点，如 *Application Config*、*IP VPN Config*、*IP Attribute Config*、*Profile Config* 等，不代表任何具体的真实设备，而是由OPNET提供来简化各种应用和网络技术的设置和配置。

2.2.2 Internet_ toolbox 调色板中的模型

存在于 *internet_ toolbox* 调色板中的模型对于多数常见的网络协议和技术的操作而言，通常是足够用的。下面给出一些这样的模型常见用途的简短描述：

1）*ppp_ wkstn* 和 *ethernet_ wkstn* 被用来对客户工作站、端节点进行建模，这两种节点分别在SLIP和以太网链路上运行各种应用。

2）*ppp_ server* 和 *ethernet_ server* 用来对服务器建模，这也是端节点，但分别在SLIP和以太网链路上为各种客户端应用提供服务。服务器和工作站对象支持客户端—服务器范型的建模。

3）*1000BaseX_ LAN*、*100BaseT_ LAN* 和 *10BaseT_ LAN* 模型被用来仿真以太网LAN，分别运行在1000BASE－X、100BASE－T和10BASE－T以太网连接上。默认情况下，上面的每种网络模型都仿真具有10个端节点的一个LAN的运行操作。

4）*ethernet32_ hub* 对一个以太网hub设备建模，这是层1重发器（repeater），连接多达32个以太网端节点，形成单个以太网分段。

5）*ethernet16_ switch* 对一台以太网交换机建模，这是具有16个接口的层2设备。

6）*ethernet4_ slip8_ gtwy* 对一台网关路由器建模，这是具有4个以太网接口和8个SLIP接口的层3设备。以太网接口通常用来连接局部子网（如LAN对象、hub或交换机），而IP接口通常用来对到IP网络和到诸如路由器、服务器和工作站等节点设备的连接进行建模。

7）*ethernet2_ slip8_ firewall* 也对具有附加防火墙功能的一台网关路由器建模。这个模型具有2个以太网接口和8个SLIP接口。

8）*ip32_ cloud* 对一个IP云进行建模，当不关注准确的拓扑时，普遍用来表示互联

网的连通性。这个模型具有 32 个串行 IP 接口。

9）*1000BaseX*、*100BaseT* 和 *10BaseT* 是以以太网接口连接设备的链路对象的模型。

10）*PPP_ 28K*、*PPP_ 33K*、*PPP_ 56K*、*PPP_ DS1* 和 *PPP_ DS3* 是具有各种数据率的双工点到点链路的模型，可连接运行 IP 的两个节点。

11）*Application Config* 和 *Profile Config* 模型用于配置被仿真网络中的应用和用户概要。分别在第 5 章和第 7 章给出有关这些模型的更多细节。

2.3 操作对象调色板树

本节给出打开 **Object Palette Tree**（对象调色板树）、寻找仿真研究所需的设备和链路模型以及创建网络拓扑的指令。

2.3.1 打开对象调色板

从 **Project Editor**（项目编辑器）内打开 **Object Palette Tree**（即一个项目场景是打开的），可使用如下两种方法中的一种方法：

方法 1：在看起来像 ▣ 的 **Open Object Palette** 图标[㊀]上单击。

方法 2：从下拉菜单中选择 **Topology→Open Object Palette**（拓扑→打开对象调色板）。

2.3.2 在对象调色板中搜索模型

Object Palette Tree 提供了一个搜索工具，帮助基于模型名寻找模型。例如，为寻找交换机设备的模型，在 *Search by name*（依名搜索）文本框中输入字“switch”，并单击 **Find Next** 按钮。搜索将从树中的当前位置开始，并将继续向下朝底部的方向搜索。如果找到模型，那么搜索工具将其高亮显示。单击 **Find Next** 按钮，在书中定位其名字中包含字“switch”的下一个模型。如果没有找到模型，那么窗口将保持不变，且没有模型被高亮显示。

不是所有的模型名字都包含描述模型性质的完整字。例如，许多链路模型在其名字中不包含字“link”，而带有以太网接口的设备仅在其名字中包含字符“eth”而不是完整的字“ethernet”。出于这个原因，应该谨慎选择搜索准则或浏览 **Object Palette Tree** 寻找关注的模型。

当在 **Object Palette** 中浏览存在的模型时，经常的情况是，对象的名字和简短描述不足以确定模型是否对当前仿真研究合适。为了得到有关所关注模型的更多细节，通过单击模型而高亮显示它，之后单击 **Model Details**（模型细节）按钮；另一种方法是，右击模型并从弹出菜单中选择 **View Model Details**（查看模型细节）选项。

㊀ **Project Editor** 的工具栏是可配置的，包含常用操作的图标。因此，可能的情况是，**Open Object Palette** 图标没有配置为可见的，在这种情形中方法 1 是不能使用的。

2.3.3 创建定制模型

有时在 **Object Palette** 中存在的模型并不包含具有必要数量或组合的所需接口的设备。出于这个目的，OPNET 允许创建自己的定制节点模型。使用如下步骤创建定制设备模型：

1）打开 **Object Palette** 窗口（见 2.3.1 节）。

2）单击 **Create Custom Model**（创建定制模型）按钮，打开如图 2.4 所示的 **Create Custom Device**（创建定制设备）窗口。

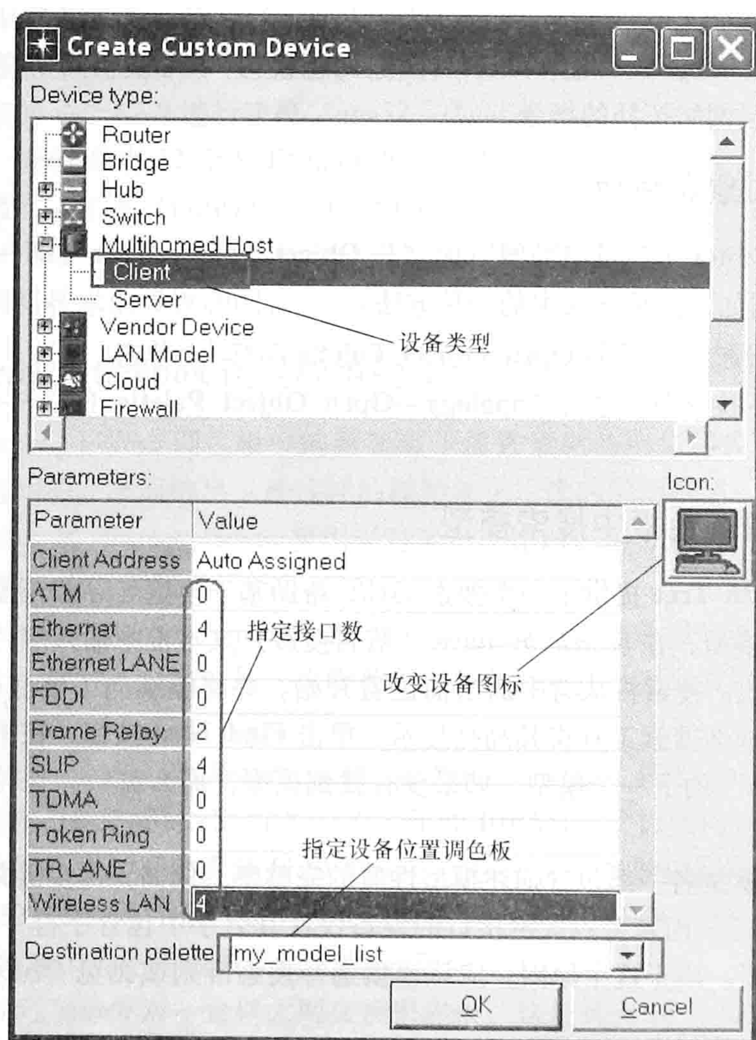


图 2.4　**Create Custom Device** 窗口

3）选择希望创建的设备类型（如路由器、网桥、交换机或云）。在一些情形中，不得不做出一个子选择，例如一台交换机可以是 ATM、以太网或 FDDI 的。

4）指定新设备参数（也称作属性）的值。这些参数通常定义这样的信息，如新设备中接口的数量和类型。某些设备会提供指定其他信息的一个选项。

5）如果必要，可改变表示新设备的图标。单击设备的图标，则打开一个 **Icon Palette**（图标调色板）窗口，使您可为新设备图标选择一个不同的图像。

6）默认情况下，新设备被存储到 *my_ model_ list* 调色板中，但可选择包含新模型的一个不同调色板，方法是从称为 *Destination palette*（目的地调色板）的下拉列表中选择一个。

7）一旦指定新设备的所有属性（包括目的地调色板），就单击 **OK** 按钮，创建该设备。

8）在提供的提示中输入新设备的名字，并单击 **OK** 按钮。OPNET 将通知您说新设备已经成功添加，并将打开新设备所驻留的文件夹中的 **Object Palette Tree**。新设备文件被存储在本地硬盘的默认目录中（通常是 op_ models）。

2.4 创建网络拓扑的方法

一旦确定并准备好必要的节点和链路设备，则就为使用 **Object Palette** 针对仿真研究创建网络拓扑做好了准备。2.4.1 ~ 2.4.4 节给出在一个网络拓扑中添加和删除节点与链路的指令以及在这些对象上执行其他编辑操作的指令。所有这些小节都假定 **Object Palette** 是打开的（见 2.3.1 节）。

2.4.1 添加节点

1）浏览或滚动 **Object Palette Tree**，并选择在仿真研究中希望使用的节点模型，方法是单击该模型。所选模型的一个图标出现在 **Object Palette Tree** 窗口的右手部分。

2）单击 **Model Details** 按钮，打开包含所选模型的一个详细描述，这在确定对象是否准确地代表了您希望仿真的设备方面是有用的。

3）为了在一个网络拓扑中创建一个新节点，将节点模型的图标拖到项目工作空间，并放到一个期望的位置。另外一种方法是，单击关注的图标，之后在项目工作空间内希望放置对象的位置处再次单击。

4）在采用上述任何一种技巧放置一个节点之后，可重复地在项目工作空间内单击以此方式添加同一对象的多个实例。

5）在项目工作空间的任何地方右击即可结束操作。

6）上述步骤可重复，重复次数与在网络拓扑中要放置各种对象类型的节点数量相同。

OPNET 也提供工具对象的模型，它们不代表任何真实的网络设备。提供这些模型是为了便利定义和配置各种网络技术和应用。向项目工作空间添加工具对象的任务也采用拖放法实施，与常规节点的方法相同。但是，不像常规节点的是，工具对象不能通过链路、需求（demand）或路径模型连接，原因是工具对象不对任何真实网络设备建模，仅针对配置目的而提供。

2.4.2 添加链路

使用如下步骤，在一个网络拓扑中的节点间添加链路：

1）在 **Object Palette Tree** 中通过单击其图标选择关注的链路模型。

2）在项目工作空间中单击希望这条链路连接的第一个节点。代表链路的一条线将出现，并将跟随鼠标移动。另一种方法是，可拖放链路模型到第一个节点，取得同样的效果。

3）将线拖到由这条链路连接的第二个节点，并单击它。这就创建了连接这两个选中节点的期望类型的一条链路。

4）可使用相同技巧，继续使用同一链路类型连接其他节点，即单击要连接的第一个节点，之后将线拖到第二个节点，并单击它。

5）通过在项目工作空间中任何地方右击，可结束这项操作。

6）有时可能希望沿一条曲线路径而不是一条直线画一条链路。为了取得这种效果，单击要连接的第一个节点。接下来，不要单击第二个节点，在项目工作空间内希望链路路径经过的一个位置单击。在定义链路路径时，可在项目工作空间内根据需要单击多次。最后，单击第二个节点完成链路路径。

2.4.3 删除节点或链路

除了将对象添加到一个网络拓扑外，有时希望从所创建的网络去除不期望的对象。如果被删除的对象是一个节点，有链路连接到该节点，那么各条链路也将被清除。有两种方式删除一个对象：

方法 1：通过单击关注的对象而选择它，之后在键盘上按 Delete 键。

方法 2：通过单击关注的对象而选择它，之后从 **Project Editor** 下拉菜单中选择 **Edit→Delete**（编辑→删除）。

2.4.4 其他编辑操作

Project Editor 支持选择一个或多个对象，并在这些对象上进行各种操作。为了选择对象，使用如下任何一种技巧都可：

方法 1：通过单击工作空间中的一个对象而选中它。之后通过 Shift - 单击或 Control - 单击更多的对象而选中它们（即在键盘上按住 **Shift** 或 **Control** 键时单击下一个对象）。

方法 2：在工作空间中任何地方单击，之后拖动光标定义一个矩形，在结束时释放鼠标。在矩形内的所有对象（节点和链路）将被选中。

在选中对象后，通过简单地用鼠标拖动它们，可将它们移动到工作空间中的一个新位置。**Project Editor** 中的 **Edit** 菜单也支持被选中对象上的如下操作，如 **Cut**（Ctrl + X）、**Copy**（Ctrl + C）、**Paste**（Ctrl + V）、**Undo**（Ctrl + Z）和 **Redo**（Ctrl + Y）。

2.5　快速配置工具

Rapid Configuration（快速配置）是创建具有大量节点的网络拓扑的一个有用工具。从 **Project Editor** 内可访问 **Rapid Configuration** 工具。

2.5.1　使用 Rapid Configuration 工具创建网络拓扑

通过如下步骤，使用 **Rapid Configuration** 工具创建一个网络拓扑：

1）从下拉菜单中选择 **Topology→Rapid Configuration...**（拓扑→快速配置）。

2）当 **Rapid Configuration** 窗口出现时，从下拉列表中选择期望的拓扑配置，并单击 **Next** 按钮，配置选中的拓扑。拓扑配置列表包括如下选项：Bus（总线）、Full Mesh（全互连）、Randomized Mesh（随机化的网状网）、Ring（环形）、Star（星形）、Tree（树形）或 Unconnnected Net（未连接的网络）。

3）**Rapid Configuration** 窗口也提供了为随机数产生指定一个种子值的选项。这项功能是非常重要的，因为一些拓扑会要求节点放置或其他性质的随机性。为指定一个种子，单击 **Seed**（种子）按钮。在 **Seed Selection**（种子选择）窗口中，输入期望的种子值，或单击 **Generate**（生成）按钮自动地为种子提供一个随机值。单击 **OK** 按钮完成选择。

4）在选择网络拓扑并单击 **Next** 按钮之后，将看到一个新窗口，这允许指定如下参数，如节点和链路模型（星形拓扑要求您为中心节点和周边节点指定模型，而树形拓扑要求您提供中间节点模型和叶子节点模型）、节点数量、节点位置和其他参数。

5）如果节点和/或链路下拉列表没有包含创建相应网络拓扑所需的模型，那么单击 **Select Models**（选择模型）按钮，并从所提供的模型列表中选择必要的技术族（technology family）。

6）一旦设置所有必要的值，单击 **OK** 按钮创建期望的网络模型，并将之放置在项目工作空间中。

2.5.2　使用 Rapid Configuration 工具创建以太局域网

作为使用 **Rapid Configuration** 工具的一个例子，这里是创建一个以太网的例子，其中使用一个星形拓扑，有 10 个以太网工作站周边节点，连接到一台以太网交换机：

1）在 **Project Editor** 中，从下拉菜单中选择 **Topology→Rapid Configuration**（拓扑→快速配置）。

2）选择 Star 作为配置拓扑，并单击 **Next** 按钮。

3）在出现的新窗口中，设置如下值：

① *Center node model*（中心节点模型）：ethernet16_ switch。

② *Periphery node model*（周边节点模型）：ethernet_ wkstn。

③ *Number*（数量）：10 是周边节点的数量。这个值不应该超过 16，原因是使用的

交换机仅有 16 个以太网接口。

④ *Link model*（链路模型）：10BaseT（这将提供 10Mbit/s 的数据速率）。

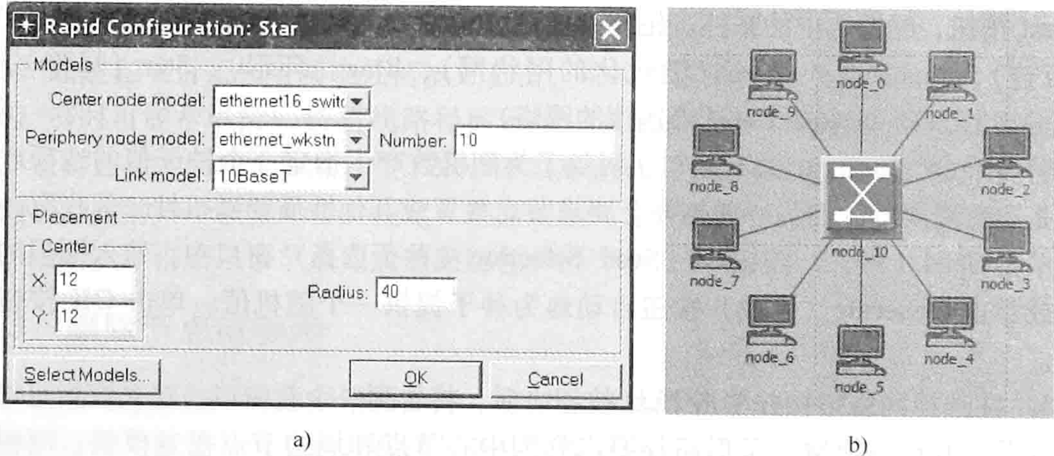
⑤ *Center*（中心）*X*：12（可使用已经提供的默认值）。

⑥ *Center*（中心）*Y*：12（可使用已经提供的默认值）。

⑦ *Radius*（半径）：40（可使用已经提供的默认值）。

4）单击 **OK** 按钮。

图 2.5 给出 **Rapid Configuration** 设置和得到的星形拓扑。如果犯了一个错误，使用错误的链路类型连接节点，或将多于那台设备上可用接口数的多个节点连接到一个设备，那么 OPNET 也许会也许不会给出有关一个无效网络拓扑的任何警告。见 2.6.2 节了解要避免的一些常见错误，以及如何验证被创建网络拓扑的有效性。



a) b)

图 2.5

a）创建一个星形拓扑的 **Rapid Configuration** 工具 b）采用这个工具创建的星形拓扑的网络

2.6 配置链路对象

链路对象连接一个网络中的节点设备。通过数据链路层（即层 2）技术定义链路，该技术核验通过链路连接的节点的兼容性。多数时间，将选择对于创建期望的拓扑合适的一个链路模型，例如，会选择链路模型 PPP_ 56K 以具有传输速率 56kbit/s 的一条点到点链路连接两个节点。但在一些情形中，在一个网络拓扑中添加一个链路对象之后，可能不得不依据需要配置链路对象。

链路模型包含各种基本属性，如传输速率或传播速度。一些这样的参数是可配置的，其他参数则不是可配置的。这取决于仿真中使用的链路模型和参数本身。包含修饰符 adv 或 int（即高级或中间模型派生层次）的链路模型通常允许改变常用参数，如链路的传输速率。但是，不管模型名字中使用的修饰符是什么，如 1000Base_ X 和 100Base_ T 等链路模型都不允许设置传输速率。

2.6.1 改变基本链路性质

可被配置的点到点链路的基本性质包括传输速率和传播延迟。使用如下步骤将如下

任意性质改变为期望的值：

1）右击链路对象，并选择 **Edit Attributes**（编辑属性）选项。

2）在 **Edit Attributes** 窗口中，在下右侧角选择 *Advanced* 选择框。

3）传播延迟以名字为 **delay** 的属性加以指定，它正常情况下设置为值 Distance Based（基于距离）。这意味着传播延迟自动地基于链路穿过的距离进行配置。如果希望独立于距离或节点的位置而将延迟设置为一个特定的值，那么单击属性的值字段，选择 Edit... 并输入期望的值。

4）传输速率以 **data rate**（数据速率）属性指定。在所有链路模型中这个属性并不都是可见的。如果当前链路模型没有包含 **data rate** 属性，那么将当前链路的 **model**（模型）属性（见 3.4.5 节）改变为具有相同名字但包含一个 int 或 adv 修饰符的另一个模型。通过在这个属性的值字段上单击，并从下拉菜单中选择 Edit...，完成这项操作。浏览出现的模型列表，并选择新的期望模型。确保新的链路模型使用相同的链路协议（即层 2 技术）。

5）通过输入以比特每秒表示的期望链路数据速率，设置 **data rate** 属性值。另一种方法是，可选择预配置的数据速率值之一（如果存在的话）。

6）单击 **OK** 按钮存储改变，并关闭 **Edit Attribute** 对话框。

2.6.2　验证链路连通性

简单地将节点放置到一个项目工作空间中并以链路连接它们，不足以创建一个网络拓扑的可正常工作模型。就节点如何采用链路进行连接有多种约束，如果不遵守这些约束，仿真将不能进行。一般而言，链路类型必须与它连接的节点类型兼容，且两侧的节点必须具有链路类型的可用接口。例如，不能将一台以太网工作站与一台以太网服务器使用一条 FDDI 或令牌环链路连接；与此不同的是，需要使用一条以太网链路。OPNET 将提供一条警告消息，如果尝试连接没有可匹配连接链路类型的可用接口的两个节点模型。如图 2.6 所示，告警消息提供两个选择：

1）*Add link and ports*（添加链路和端口）——节点模型将被更新包括一个接口（如端口），这允许一条连接使用选中的链路。

2）*Add link without assigning ports*（在不指派端口的条件下添加链路）——节点模型将不被更新，结果是，添加的链路将变得不可工作（nonfunctional）。

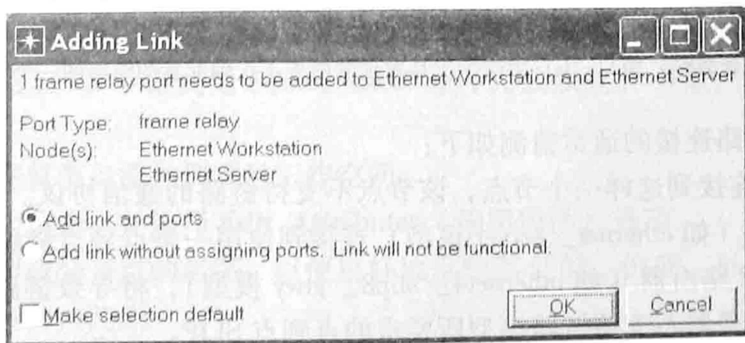


图 2.6　在添加一条不兼容的链路时的告警消息

存在其他情况，其中OPNET没有给出有关不正确连接的链路的一条警告消息，这会产生一个不正确的网络拓扑（例如，如果尝试改变一个已经连接的节点或链路对象的 **model** 属性时的情况）。出于这个原因，一旦网络拓扑被创建，验证网络中的所有节点都是正确连接的就是一个好想法。OPNET提供了实施这项任务的一个简单工具，称作 **Verifying Link Connectivity**（验证链路连通性）。下面是使用这个工具的步骤：

1）从下拉菜单中，选择 **Topology→Verify Links**（拓扑→验证链路），或另外一种方法是，按下 Ctrl + L 键。这将打开 **Check Links**（检查链路）窗口（见图2.7a）。

2）在这个窗口中，选择 **Verify links**（验证链路）选项并单击 **OK** 按钮。也可单击 **Help** 按钮，查找有关 **Check Links** 窗口的其他信息。

如果所有链路都是正确连接的，则在 **Project Editor** 窗口底部将出现“All links and paths are connected properly”（所有链路和路径都正确连接）消息。否则，如果两条不正确连接的链路被识别，那么这两条链路将被标记为红色X，且在 **Project Editor** 窗口底部的消息将是“2 incorrect links and paths were found”（找到两条不正确的链路和路径）。图2.7b形象地给出这样一个情形。

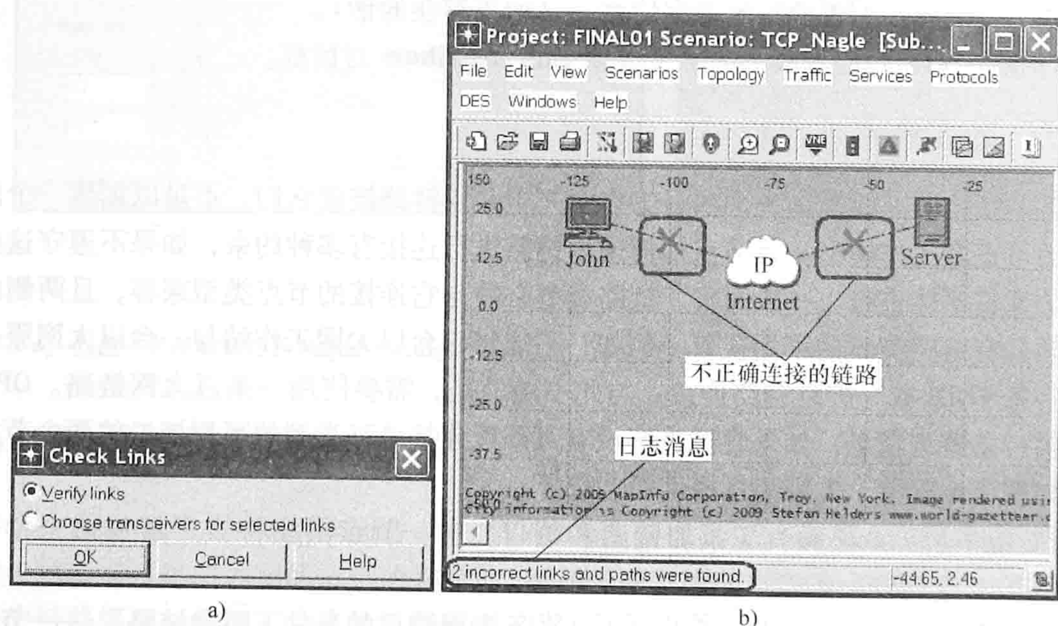


图 2.7

a）由 **Verifying Link Connectivity**（验证链路连通性）工具打开的 **Check Links**（检查链路）窗口 b）由这个工具发现的两条不正确配置的链路的一个网络

不正确的链路连接的通常猜测如下：

1）将链路连接到这样一个节点，该节点不支持链路的通信协议。例如，直接将一台以太网服务器（如 ethernet_ server 模型）连接到使用一条点到点链路（如 PPP_ DS1 链路）的一台IP路由器（如 ethernet4_ slip8_ gtwy 模型），将导致链路成为不工作的，原因是以太网服务器不支持链路模型所要求的点到点SLIP。

2）将多于一台设备上可用通信接口数的节点连接到该设备上。例如，将 20 台以太网服务器连接到一台具有 16 个接口的以太网交换机（如 ethernet16_ switch 模型）将导致 4 条连接链路成为不可工作的，原因是该交换机仅支持 16 个接口，而多余的 4 台服务器将没有接口可连接。

3）将一条链路连接到错误的端口。如路由器（如 ethernet4_ slip8_ gtwy 模型）的一些设备支持多种接口类型。因此，即使该路由器支持 IP 接口，但将一条点到点链路连接到以太网端口也将导致该链路成为不可工作的。一般而言，如果可用的话，则 OPNET 自动地为要创建的连接选择正确的接口。但是，不正确的链路到端口指派仍然会发生在某些情形中。例如，当使用 **Rapid Configuration** 工具时（见 2.5 节）的情况。

2.7　使网络单元失效并恢复

OPNET 提供了在仿真过程中使网络单元失效和恢复的工具。经常要做到如下判断是重要的，当一条或多条链路宕掉时，或如果一个关键节点（如一台服务器或路由器）失效的话，确定系统的行为如何。明显的是，人们可简单地去除将失效的单元，运行仿真并收集结果，之后将去除的单元添加回来。但是，去除且之后将对象加回到网络拓扑这种操作，会是一件非常烦琐的任务。另外，去除且之后加上网络单元不支持研究这样的一种情况，其中链路或节点对象失效，之后在仿真中间恢复该对象。例如，当希望知道在单元失效或恢复之前和之后网络及其协议的行为如何时的情况。为了解决这样情形，OPNET 提供了在网络中使单元失效和恢复的一个独立的工具节点。在网络中使对象失效/恢复有两种方法：

方法 1：在整个仿真中使对象失效/恢复。

1）选择要失效/恢复的对象，方法是单击这些对象（如果要选择多个对象，Shift/Control－单击）。

2）从下拉菜单中，选择 **Topology→Fail Selected Objects**（拓扑→使选中的对象失效）或 **Recover Selected Objects**（恢复选中的对象）。另外一种方法是，单击图标使对象失效，单击图标恢复选中的对象。

方法 2：在仿真过程中的特定时刻使对象（可能是多个）失效/恢复。

1）打开 **Object Palette**（对象调色板）。

2）以此展开 *Shared Object Palettes*（共享对象调色板）组、*utilities*（工具）和 *Node Models*（节点模板），其中将找到 *Failure Recovery*（失效恢复）节点。也可直接浏览查找这个对象。

3）将失效恢复节点添加到项目工作空间。

4）在节点上右击，并选择 **Edit Attributes**（编辑属性）选项。

5）可能希望改变节点的名字，以便更好地反映其目的。例如，可将这个节点命名为 Failure 和 Recovery 节点。

6）为了使链路失效/恢复，展开它的 **Link Failure/Recovery Specification**（链路失

效/恢复规格）属性。

7）对于每个失效或恢复事件，在相应属性中添加一行。例如，为了使名为node_ 0的节点在时刻150s时失效，并之后在时刻200s时恢复，需要将在 **Node Failure/Recovery Specification** 属性中的行数设置为2。

8）展开每行，并设置如下属性值：要失效或恢复的节点或链路的 **name**（名），失效/恢复的 **time**（时间）和 **status**（状态）[为Fail（失效）或Recover（恢复）]。

9）当使节点失效/恢复时，也可能希望设置 **Node Failure Mode**（节点失效模式）属性。该属性有如下值：

① Node and attached Links（节点和连接的链路）——使节点及连接到节点的链路失效/恢复。

② Node Only（仅节点）——仅使节点失效/恢复，而链路保持不受影响。

③ Attached Links Only（仅连接的链路）——仅使连接到节点的链路失效/恢复。

10）单击 **OK** 按钮存储改变。

图2.8给出 *Failure Recovery*（失效恢复）节点的样例设置，其中在改变了值的属性周围带有环形框。使用上面的配置，node_ 0将在时刻150s处失效，并之后在时刻200s处恢复。注意：**Node Failure Mode**（节点失效节点）属性将设置为Node Only的默认值。

2.8 子网

OPNET产品为将网络拓扑的各网元分组为独立的子网包含了一个非常有用的功能特征。OPNET为持有其他对象（如节点、链路和其他子网对象）提供了一个特殊的子网对象。在子网对象的帮助下，可将网络中的各节点以一种层次结构的方式安排布局。如果子网A包含子网B，那么子网A被称为子网B的父子网，而子网B是A的一个孩子子网。通过一个或多个子网对象可指定层次结构的多个层次，每个层次代表了较大型网络的一个物理或逻辑部分。子网层次结构的最高层被称作顶层或全局子网。划分子网的做法为以一种便利对网络不同部分的快速和容易的访问和查看的方式，提供组织复杂网络的一个强大工具。

当处理大型网络时，将拓扑组织成为子网通常是工作的第一要务。首先，需要为拓扑中将有的每个子网添加一个子网对象。接下来，应该以诸如节点、链路的网络单元（也许有其他子网），在被创建的每个子网中放置这些网元。工具对象负责整个仿真研究的配置，由此应该仅添加一次，通常在网络的顶层或在顶层子网的孩子子网之一中添加。

2.8.1 添加一个子网对象

方法1：从 **Project Editor**（项目编辑器）中。

1）从 **Project Editor** 的下拉菜单中，选择 **Topology→Subnets**（拓扑→子网）。

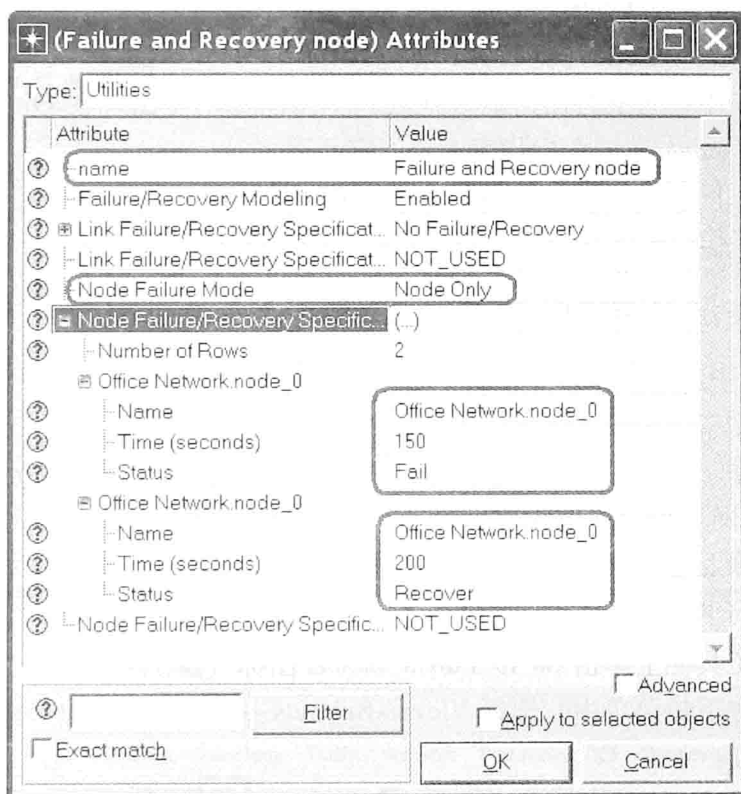


图 2.8　*Failure Recovery* 节点的样例属性设置

2）从子菜单中，选择如下情形之一：

① **Create Fixed Subnet**（创建固定子网）——创建一个物理子网，其中在放置过程中各节点不允许移动。固定子网代表静态网络，通常通过一条或多条物理链路连接到其他节点或子网。

② **Create Logical Subnet**（创建逻辑子网）——创建一个逻辑或虚拟子网，不考虑诸如地理节点位置等物理拓扑特征。逻辑子网也通常通过一条或多条链路连接到其他节点或子网。

③ **Create Mobile Subnet**（创建移动子网）——创建一个移动子网，其中在仿真过程中允许各节点移动。移动子网以及放置在其内部的节点不允许通过物理链路连接到其他节点或子网。

④ **Create Satellite Subnet**（创建卫星子网）——创建一个卫星子网。像移动子网一样，卫星子网以及放置在其内部的各节点不允许通过物理链路连接到其他节点或子网。

3）在希望放置子网对象的项目工作空间内部的位置单击。OPNET 的一些版本会询问您指定新添加对象的名字。

4）如期望的添加足够多的子网对象，并在项目工作空间的任何地方右击来结束该过程。

方法 2：从 **Object Palette**（对象调色板）中。

1）打开 **Object Palette**（见 2.3.1 节）。

2）在项目工作空间中的任何地方拖放，子网图标之一出现在 **Object Palette** 窗口的右侧。注意名为 *Subnet* 的图标代表一个固定子网。

3）通过在项目工作空间中再次单击的做法，可添加一个以上的选中类型的子网。

4）在项目工作空间的任何地方右击，结束添加选中子网的操作。

2.8.2　在网络层次结构中移动对象

1. 进入一个子网（从层次结构向下进入）

方法 1：双击子网图标。

方法 2：右击子网图标，并从弹出子网图标菜单中选择 **Enter Subnet**（进入子网）。

2. 进入父子网（沿层次结构向上移动）

方法 1：在项目工作空间的任意地方右击，并从弹出菜单中选择 **Go to Parent Subnet**（进入父子网）。

方法 2：在工具栏上单击 *Go to Parent Subnet* 图标（）。

方法 3：从主下拉菜单中，选择 **View→Subnets→Go to Parent Subnet**（查看→子网→进入父子网）。

方法 4：按 Ctrl + 0（数字零）键。

例如，考虑如图 2.9 所示的网络拓扑。图 2.9a 给出称为 IP 的项目的 cloud_ modeling 场景（如窗口标题所示）中的顶层网络拓扑。图 2.9a 中给出的拓扑由分别称为 client_ site1、…、client_ site5 的五个子网组成，都连接到对互联网建模的节点。

注意在图 2.9a 中，*Go to Parent Subnet* 图标是不使能的，原因是在顶层之上没有子网。图 2.9b 形象地给出同一场景的 client_ site3 子网，也由窗口标题指明了。注意在右侧图像中，*Go to Parent Subnet* 图标是激活的。在 client_ site3 子网的项目工作空间中单击那个图标，将回到父子网，这是如图 2.9a 所示的顶层子网。

2.8.3　使用子网创建一个网络拓扑

1）在开始的（starting）网络层，依据需要，添加节点、链路和工具对象。

2）如果当前的网络层在其结构中包含子网，那么为每个子网添加一个子网对象。

3）为每个子网对象建立网络拓扑：

① 进入子网。

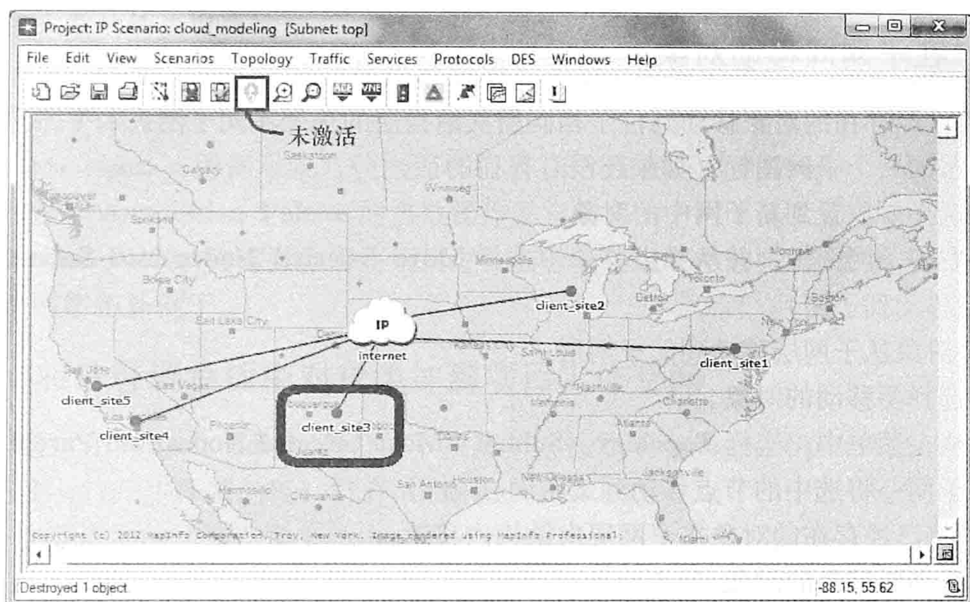
② 通过从 **Object Palette**（对象调色板）（见 2.3 节）拖放对象，或通过使用 **Rapid Configuration**（快速配置）工具（见 2.5 节），在子网内创建网络拓扑。

4）将一个父子网中的对象连接到孩子子网内的对象：

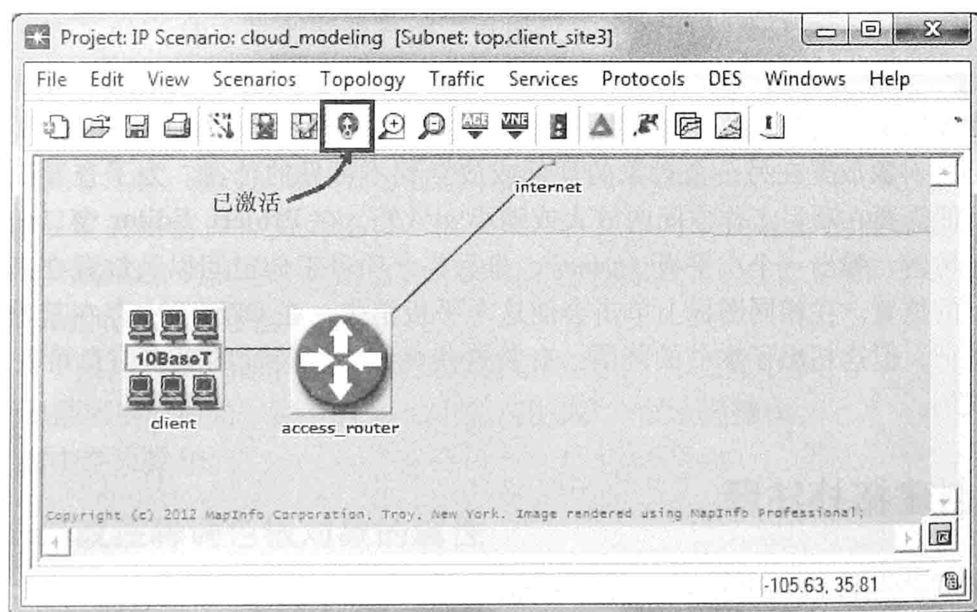
① 进入父子网。

② 添加一条链路，将父子网中的期望节点与孩子子网连接。

③ 将一条链路连接到一个子网，就会打开 **Select Node**（选择节点）窗口，其中列



a)



b)

图 2.9　带有子网的一个网络拓扑例子

a）顶层子网　b）client_ site3 子网

出了孩子子网中包含的所有节点。

④ 从 **Select Node** 窗口的下拉列表中，选择希望连接到的节点，并单击 **OK** 按钮。

为了将一个子网内的一个节点与父子网中的另一个节点连接，需要首先进入父子网，仅有此时才可添加期望的链路。类似地，要连接在两个孩子子网中的节点，需要首先进入共同的顶级父子网，仅有此时才可添加一条链路，连接在相应孩子子网中的节点。

2.8.4 在子网间移动对象

为将已经存在的对象移动到位于相同层次结构层的一个不同子网：

1）添加一个子网图标（如果还没有存在的话）。

2）选择要放置到新子网中的对象。

3）右击子网图标，并从弹出菜单中选择 **Move Selected Nodes into Subnet**（将选中的节点移动到子网）。

为将对象从子网层移动到父子网层：

1）选择要移动的对象。

2）从主菜单中，选择 **Topology→Subnet→Move Selected Nodes into Parent Subnet**（拓扑→子网→将选中的节点移动到父子网）。

为了将已经存在的对象在子网层次结构内移动：

1）选择要移动的对象。

2）从主菜单中选择 **Topology→Subnets→Move Selected Nodes into Specific Subnet**（拓扑→子网→将选中的节点移动到特定子网）。

3）从 **Choose Subnet**（选择子网）窗口，选择希望将节点移到的子网，并单击 **OK** 按钮。

当在子网层次结构内移动对象时，应该小心谨慎，原因是 **Project Editor**（项目编辑器）会将对象放置在另一个对象的顶部或放置到不可见的位置。为了查看被移动的对象，可能需要在项目工作空间内放大或缩小。另外，在 **Project Editor** 窗口的右下角的图标上单击，弹出一个小平板（pane），显示整个项目工作空间以及放置在其内部的所有节点的位置。在相同图标上单击会使这个平板消失。在 OPNET 中存在其他高级子网功能特征，但这超出了本书的范围。有关这些高级功能特征的更多信息可在 OPNET 文档中找到。

2.9 创建拓扑注释

OPNET 产品包括一个注释工具，可帮助更好地组织网络拓扑，并提供各种可视（visual）的和文本性的提示。注释工具使您可标注网络中的对象，使用不同颜色画线、圆和矩形，并以选中的颜色填充所画的对象。

如图 2.10 所示，**Annotation Palette**（注释调色板）由如下 5 个对象组成，每个对象均可被放置到项目工作空间中：



图 2.10 注释调色板

1）*Box and ellipse objects*（方块和椭圆形对象）使您在项目工作空间中画出矩形和椭圆形。这些形状可设计带有不同颜色、尺寸和位置、填充颜色、旋转和其他特点。

2）使用 *Line objects*（线对象）画出具有各种属性的直线，如颜色、位置、在直线

的始端或末端的箭头以及线型［如实线或点画线（dashed）］。

3）*Text objects*（文本对象）使您可在项目工作空间中添加条目的文本描述。它们的属性包括字体大小、文本颜色、位置和其他特点。

4）*Icon objects*（图标对象）使您可将任意 OPNET 图标放到 **Project Editor**。

虽然操作 **Annotation Palette** 的多数操作是自解释的，但 2. 9. 1 ~ 2. 9. 3 节还是提供了 **Annotation Palette** 关键操作的简短描述。实验室作业 2 中的图 L2. 1 给出带有注释对象的一个网络拓扑例子。

2. 9. 1　将注释调色板中对象添加到项目工作空间

1）通过在 **Project Editor** 窗口的下拉菜单中选择 **Topology→Open Annotation Palette**（拓扑→打开注释调色板）打开 Annotation Palette。

2）在要添加的注释对象上单击：

① 对于方形（box）和椭圆形对象，将对象的轮廓拖到项目工作空间中。

② 对于直线对象，在项目工作空间中希望直线开始的地方单击。接下来，在一个点单击会结束直线，并在同一点开始另一条直线分段，双击左键终止直线，或右击取消画直线。可依据需要添加多条直线分段，方法是在以一次双击左键结束操作之前，在工作空间中多次单击。

③ 从版本 14. 0 开始，当添加一个文本对象时，OPNET 自动地为输入文本打开编辑器。一旦输入文本，通过从编辑器下拉菜单选择 **File→Commit**（文件→提交）（或在右上角单击按钮关闭窗口）关闭窗口。接下来，将文本对象的轮廓拖到项目工作空间（要将文本对象添加到其中）内的位置。OPNET 的较陈旧版本会首先询问您指定对象轮廓，且仅有此时才允许您输入文本。

④ 对于图标对象，简单地在希望放置图标的项目工作空间中单击。

3）如果期望添加同一类型的多个对象，则多次重复上述操作。

4）右击结束操作。

2. 9. 2　修改注释调色板对象的属性

为了修改项目工作空间中任意 **Annotation Palette** 对象的属性，右击关注的对象，并从弹出菜单中选择 **Edit Attributes**（编辑属性）。

1）针对方形和椭圆形对象，频繁使用的属性如下：

① **Color**（颜色）指定对象的线条颜色。

② **Fill**（填充）指定对象的填充颜色。

③ **Width/Height**（宽度/高度）指定对象的尺寸。

④ **Rotation**（旋转）通过指定的度数，可旋转对象。

2）针对线条对象，频繁使用的属性如下：

① **Color**（颜色）指定线条的颜色。

② **Drawing style**（画线类型）通过将这个属性的值设置为 spline，可平滑线条

分段。

③ **Line style**（线条类型）将对象画为实线或点画线。

④ **Head arrow/tail arrow**（头端箭头/尾部箭头）在线条的始端或末端添加一个箭头。

3）针对文本对象，频繁使用的属性包括如下：

① **Color**（颜色）指定文本的颜色。

② **Font**（字体）指定文本的大小。

③ **Background color**（背景颜色）设置文本在其内放置的方形的填充颜色。

④ **Rotation**（旋转）通过指定的度数，可旋转对象。

4）针对 *icon*（图标）对象，频繁使用的属性如下：

Icon name（图标名）指定在项目工作空间中显示的图标。不能直接输入名字，但如果单击这个属性的值，就会打开一个窗口，可选择中意的图标。

2.9.3 在项目工作空间中显示/隐藏注释调色板对象

为了从 **Project Editor** 的下拉菜单中显示/隐藏所有注释对象，选择 **View→Annotations→Show in Subnet**（查看→注释→显示在子网中）。这会双态打开关闭（toggle）注释对象，交替地隐藏和显示它们。

2.10 去除节点杂乱放置

OPNET 提供了几项功能，使得可在包含大量图标的网络拓扑中去除杂乱放置和重叠。下面描述每项功能，并提供使用功能的指令：

1）**Automatic Icon Scaling**（自动图标缩放）；当图标相互放置得太近，且它们的图像开始重叠时，自动地调整图标的大小。为了激活/禁止这项功能，从 **Project Editor** 的下拉菜单中选择 **View→Layout→Automatic Icon Scaling**（查看→布局→自动图标缩放）。通过改变名为 **Disable Icon**（禁止图标）的首选项的值，也可通过 **Preferences Editor**（首选项编辑器）（见 1.2 节），设置自动图标缩放。

2）**Autosizing**（自动调整大小）自动地调整工作空间中图标大小。这项功能会负面地影响仿真性能，由此当场景中的图标数超过某个阈值时，这项功能会自动地关闭。通过修改名为 **Element Count Threshold for Aggressive Icon Autosizing**（聚合图标自动调整大小的单元计数阈值）的值，可通过 **Preferences Editor**（首选项编辑器）改变这个阈值。这个首选项的默认值是 500。

3）**Automatic Label Placement**（自动标签放置）是降低被创建网络拓扑杂乱放置的另一项功能，方法是自动地放置对象标签，从而使它们相互不会覆盖。为了激活这项功能，选择 **View→Layout→Automatic Label Placement**（查看→布局→自动标签放置）。通过如下标记改变首选项的值，也可通过 **Preferences Editor**（首选项编辑器）（见 1.2 节）控制这项功能：

① **title_ autoplacing. try_ small_ font**——使用较小的标签字体来降低重叠。

② **title_ autoplacing. directions**——在尝试降低重叠过程中，在不同位置（如顶部、底部、左侧或右侧）放置标签。

③ **title_ autoplacing. disable**——控制自动标签放置是否激活。

④ **Interactive Icon Scaling**（交互式图标缩放）使您可手工地调整项目工作空间中图标的大小。如果没有在工作空间中选择任何对象，那么在当前场景中的所有图标都将受到影响；否则，图标缩放仅应用到选中的对象。为了使用这项功能，选择 **View→Layout→Scale Node Icons Interactively**（查看→布局→交互地缩放节点图标）。

第3章　配置网络拓扑

3.1　引言

在第2章描述的建立一个网络拓扑，仅是开发一项仿真研究中的第一步且通常是最简单的一步。后续步骤包括如下任务，如配置个体链路和网络设备，设置并部署协议和应用，以及定义网络中用户的概要（profiles）等。在OPNET中，每个网元被表示为一个对象。网元的特点或属性是通过对象 **attributes**（属性）指定的。通过修改仿真模型中相应对象的属性值，实施网络设备配置。一项仿真研究的准确度不仅取决于各种节点及其连接链路的放置如何精确地匹配真实生活中的网络，而且取决于被创建网络拓扑中设备的正确配置。OPNET创建的被仿真网络表示越接近于真实网络，则仿真结果的准确度就越高。OPNET提供设备和链路模型，支持大多数常用的网络协议。但是，不需要指定仿真研究中所有协议和技术的所有属性的值，因为每种协议通常都预先配置有最频繁使用的默认值。多数时间，仅需要改变那些默认值中的一些值。

例如，如果希望开发一项仿真研究，考察TCP最大分段尺寸（MSS）值对一项文件传输协议（FTP）应用性能的影响，那么将需要创建代表要研究的网络的一个拓扑，配置并部署FTP应用，并仅在网络中被选中节点的TCP配置中改变MSS参数的默认值。另外，如果希望比较各种路由协议的性能，那么仿真研究会要求几个场景，每个场景都采用有一个不同的路由协议。在这种情形中，可将仿真配置有部署路由信息协议（RIP）的一个场景，采用中间系统到中间系统（IS－IS）协议的第二个场景，采用开放最短路径优先（OSPF）的第三个场景，等等。在这种情形中，虽然从一个场景到另一个场景路由协议改变了，但也许不需要改变仿真中使用的设备模型。这种情况是非常常见的，原因是设备模型通常支持可互换使用的各种协议，如路由协议RIP和OSPF、传输协议TCP和用户数据报协议UDP。结果，经常可通过简单地修改设备配置而改变一个场景中的协议。

OPNET软件是非常复杂的，且它通常允许不同的替代过程完成相同的任务。指定协议并配置它们的属性经常可使用不同指令集实施。本章主要集中在配置网元的指令，方法是改变它们的属性值。具体而言，本章描述对象属性的不同类型，解释如何以各种方式在各种网元中设置属性值，并提供一些常用设备属性的范例。在讨论每个协议层的后面各章中，描述配置的协议特定的各种过程。

3.2　对象属性

改变一个对象的属性设置的最简单方式是修改其属性的值。本节描述一个对象会有

的属性的类型以及如何在任何特定对象中访问这些属性。

3.2.1 属性类型

一般而言，任意对象可能拥有的属性有三种类型：

1）**Basic**（基本的）或 **noncompound attributes**（非复合属性）代表相应对象的单一性质。一个非复合属性没有子属性，且有指派给它的单一值。非复合属性的例子是缓冲尺寸，表示为一个整数值，表示为一个字符串的 IP 地址，表示为双精度浮点数的应用开始时间，等等。

2）**Compound attributes**（复合属性）依据一些常见特点分组子属性。复合属性包含一个或多个子属性，这些子属性可以是基本的或复合的。一个复合属性也可有指派给它的一个值。一个复合属性的值是其所有子属性值的一个集合。一个复合属性值通常有与之关联的一个名字。例如，属性 **TCP Parameters**（TCP 参数）由几个子属性组成，如接收缓冲尺寸、MSS、最大确认（ACK）延迟、快速重传特征的可用性、快速恢复特征的可用性等。指派给所有这些子属性的值集合代表复合属性 **TCP Parameters**（TCP 参数）的值。为了简化 TCP 的配置，OPNET 提供了 **TCP Parameters** 属性的预配置的值。一些这样的值具有诸如 Default、Tahoe、Reno、New Reno、SACK、NT（3.5/4.0）等的名字，这对应于一个 TCP 版本/风格，其选择自动地确定子属性值的设置。另外的方法是，可选择分别指定子属性的值。

3）**Grouping attributes**（分组属性）或简单说 **attribute groups**（属性组）类似于复合属性，区别在于分组属性没有与之关联的值。**attribute groups** 的唯一作用是将具有共同目的或属于同一协议的那些属性组织到单一分类。例如，属性组 **IP Routing Protocols** 由几个复合子属性组成，每个子属性代表 IP 路由协议之一的配置，这些协议如边界网关协议（BGP）、内部网关路由协议（IGRP）、增强的内部网关路由协议（EIGRP）、IS－IS 等。属性组 **IP Routing Protocols** 不能有指派到它的值。

3.2.2 对象弹出菜单

放置到项目调色板中的任何对象（包括网络设备、链路、需求、域对象、工具节点、各种注释条目等）都有与之关联的一个 **Object Pop－up Menu** 菜单。在对象上右击打开这个菜单。它包含在这个对象上频繁实施的一个选项或动作集合，包括称为 **Edit Attributes** 的一个选项，这使您可修改对象的属性值。

取决于对象类型，在 **Object Pop－up Menu** 中的可用动作列表是不同的。例如，链路对象的一个操作是重新定义链路的路径，但这项操作对设备或工具节点是不可用的。下面给出在 **Object Pop－up Menu**（对象弹出菜单）（见图 3.1）中可用的最常用操作的描述：

1）**Edit Attributes**（编辑属性）——这是打开对象的 **Edit Attributes** 对话框（默认视图）的关键操作之一，这使您可查看并修改对象的主要属性（即性质）。**Edit Attributes** 对话框中的默认视图仅显示主要的或频繁使用的属性。但是，对话框包含称为

Advanced（高级）的检查框，这使您可在对象属性的高级视图和默认视图之间切换。

2）**Edit Attributes**（**Advanced**）——在高级视图激活的情况下打开对象的 **Edit Attributes**。高级视图包含主要属性和高级属性。在 **Edit Attributes** 对话框的默认视图中高级属性是不可用的。在高级视图中去选中 *Advanced*（高级）检查框将切换回默认视图。

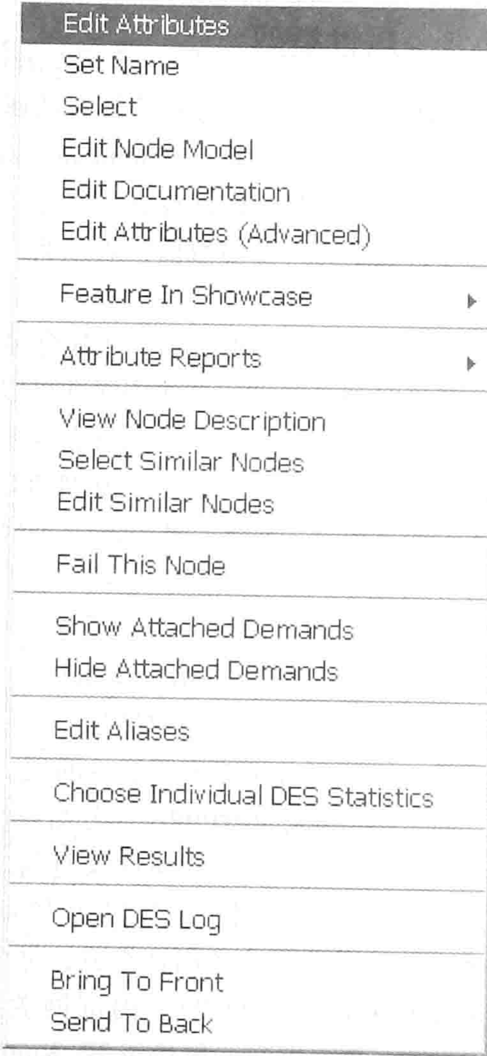


图 3.1　对象弹出菜单

3）**Set Name**（设置名字）——这项操作使您可改变对象的名字，使仿真研究更加对用户友好。通过改变 **Edit Attributes** 对话框中 **Name** 属性的值，也可改变对象的名字。

4）**View <Object> Description**（查看<对象>描述）——打开包含对象模型的一般描述的一个窗口。对象可以是一个节点、链路、需求等。

5）**Edit <Object> Model**（编辑<对象>模型）——这个选项仅在 OPNET Modeler 中是可用的。它打开对象的节点视图，并使您可编辑节点的模型。

6）**Select**（选择）——选择或高亮当前对象。通过简单地单击对象，可取得相同效果。

7）**Select Similar <Objects>**（选择类似<对象>）——选择相同类型的所有对象（即那些具有相同仿真模型的对象）。对于同时改变多个对象中的属性值，这个选项是有用的（见 3.4.2 节）。

8）**Fail This <Object>**（使这个<对象>失效）——将当前对象标记为失效，从而使它在仿真过程中是不可用的。可被标记为失效的对象包括节点、链路和需求。OPNET 不允许对子网对象实施失效操作。

9）**Recover This <Object>**（恢复这个<对象>）——如果当前对象以前被标记为失效，那么通过使之在仿真过程中是可运行的，这个选项就恢复了该对象。可被恢复的对象包括节点、链路和需求。OPNET 不允许恢复子网对象。

如果在包含多个对象的一个项目调色板位置右击，那么 OPNET 提供一个不同的弹出菜单，列出在那个位置放置的所有对象。之后可选择要修改的对象。图 3.2 给出这样一个情况，其中右击包含四个需求对象的一个点，这就弹出列出这些需求的一个菜单。之后该图给出在列表中被选中进行编辑的第一个需求。



图3.2　从 **Multiple Object Pop – up Menu**（多个对象弹出菜单）中选择一个对象

3.3　编辑属性对话框

Edit Attributes 对话框可从 **Object Pop – up Menu**（对象弹出菜单）中访问，列出描述 OPNET 模型性质的各属性，其中包括在对象内部可用的各项网络协议和技术的配置参数。对话框是以具有三列的一个表的形式组织的：

1）*Attribute Description*（属性描述）：对话框的最左列是空的，或包含一个图标（是一个圆圈，内部有一个问号）。单击这个图标，打开一个窗口，其中有属性的一个简短描述。如果该列没有为一个属性包含一个图标，那么就不存在那个属性的描述。将属性分组通常没有与它们关联的任何属性描述。

2）*Attribute Name*（属性名）：中间列包含属性名。每个属性可以是3.2.1节中描述的三种类型中的任何一种，即基本、复合或分组属性。在一个树视图中给出了属性分组和复合属性，其中叶节点是具有将值指派给它们的基本属性，而非叶节点是复合属性或分组属性。复合属性和分组属性可被展开查看它们的子属性，或被收缩给出对象属性的一个比较紧缩的视图（见3.3.2节）。

3）*Attribute Value*（属性值）：最右侧列显示属性值。通常情况下，多数对象都会有预配置某些默认值的属性。但是，可改变默认属性值，以便比较准确地反映希望建模的系统。

图3.3a给出 **Edit Attributes**（编辑属性）对话框的一个默认视图。如图所示，在对话框的顶部，有一个非复合属性 **name**（名），它定义了这个对象的名字。属性 **Reports**（报告）是一个分组属性，因此没有值指派给它。相反，复合属性 **TCP Parameters**（TCP参

数）有指派给它的一个预设值 Reno。这个值表明节点被配置支持 TCP 的 Reno 风格(flavor)。

图 3. 3a 也给出两个检查框，在右下角附近高亮显示一个矩形。选择检查框 *Advanced*（高级），就提供附加的过滤选项，并使所有可用的属性都是可见的（见图 3. 3b）。选择检查框 *Apply to selected objects*（应用到被选中的对象），就将属性改变应用到已经选中的所有对象。这是在多个对象中改变一个特定属性的值的一种方便的方法，但这要求在所有对象中将属性设置为相同的值。在打开 Edit Attributes（编辑属性）对话框之前，必须预先选择对象。如果由于某种原因，一个被选中对象不能被改变，那么 OPNET 将不改变值，但它也不会提供任何警告消息。验证改变已经应用到所有被选中的对象，是您的责任。同样，如果改变了作为一个复合属性组成部分的任意属性，那么在所有被选中对象内，整个复合属性值将被替换，这可能导致不期望的结果。

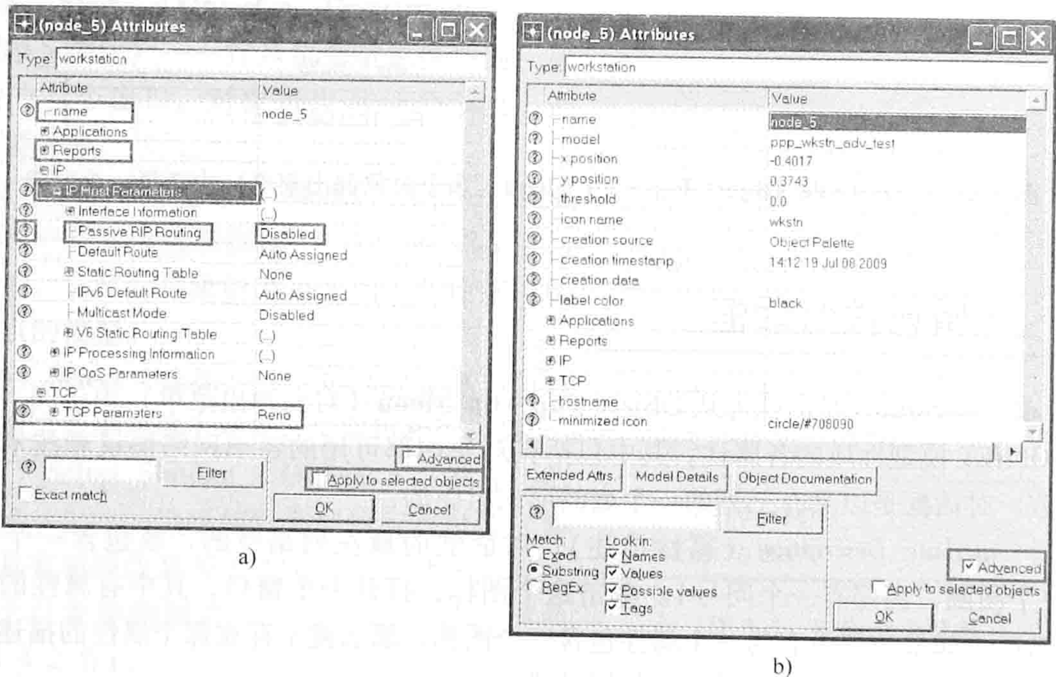


图 3. 3 Edit Attributes（编辑属性）对话框

a）默认视图 b）高级视图

3. 3. 1 访问属性描述

使用如下步骤，查看一个属性的详细描述：

1）右击关注的对象，并选择 **Edit Attributes** 选项，就打开 **Edit Attributes** 对话框。

2）浏览对话框中存在的属性，找到关注的属性。

3）在 Edit Attributes 对话框的最左列，基本属性和复合属性有一个图标（形式为一个圆圈，内部有一个问号）。单击图标，打开带有被选中属性的描述的一个窗口。

3.3.2　操作复合和分组属性

在 **Edit Attributes** 对话框中没有展开的复合和分组属性，以内部有加号的一个图标标记。为了展开这样一个属性，使用如下方法之一：

方法1：单击在属性名左侧的加号或小箭头。这导致属性被展开，并将图标改变，从而现在它包含一个减号。

方法2：在一个复合属性的值字段上单击可查看和/或选择可用的配置选项之一。选择选项 Edit...，将展开当前属性，打开一个弹出窗口，其中可配置当前复合属性的所有子属性。在一些复合属性中，在值字段上单击，将直接打开带有子属性的弹出窗口。这种方法不适用于分组属性，原因是它们没有值。

方法3：在一个复合或分组属性的值字段上右击，并选择选项 **Expand Row**（展开行）。通过在属性的名字字段上右击，也可完成相同的操作。一般而言，在一个复合属性上右击，提供如下选项（对于一个分组属性，仅有 **Expand Row** 选项是可用的）：

① **Expand Row**（展开行）——展开复合属性。

② **Promote Attribute To Higher Level**（将属性提升到较高层）——使您可在仿真层配置这个属性。参见3.5节了解更多细节。

③ **View Attribute**（查看属性）——打开具有属性配置的一个只读弹出窗口。

例如，如图3.3所示，当展开时，分组属性 **IP** 包含三个附加复合子属性：**IP Host Parameters**（IP 主机参数）、**IP Processing Information**（IP 处理信息）和 **IP QoS Parameters**（IP QoS 参数）。复合属性 **IP Host Parameters** 也被展开，且它包含几个复合和基本子属性。**Passive RIP Routing**（被动 RIP 路由）是 **IP Host Parameters** 的一个子属性。这是一个基本子属性，其值设置为 Disabled（禁止）。

3.3.3　具有多个实例的属性

某些对象可能包含同一属性的多个实例。例如，一台服务器可支持多项应用服务，每项服务都需要单独配置。在这样一种情形中，正被谈论的一个属性可能有称为 **Number of Rows**（行数）的一个孩子（即子属性）。属性 **Number of Rows** 的值对应于要创建的父属性的实例数。默认状况下，对于多数实例，**Number of Rows** 的值是0。如果那个值发生改变，那么将在 **Number of Rows** 属性的同一层添加一个新的属性集合。图3.4给出要求多个实例的一个属性。

如图3.4a所示，最初情况下，属性 **Number of Rows** 被设置为0。一旦该属性的值改变为2，则出现两个新的复合属性（见图3.4b）。默认状况下，添加属性的名字是 **None**，且它与 **Profile Name**（概要名）属性的值是相同的。在 **Profile Name** 属性的值被改变为 Browsing 之后，被添加复合属性的名字也变为 **Browsing**。当处理同一属性的多个实例时，常见的情况是，一个新添加的属性实例将其名字改变为其子属性之一的值。

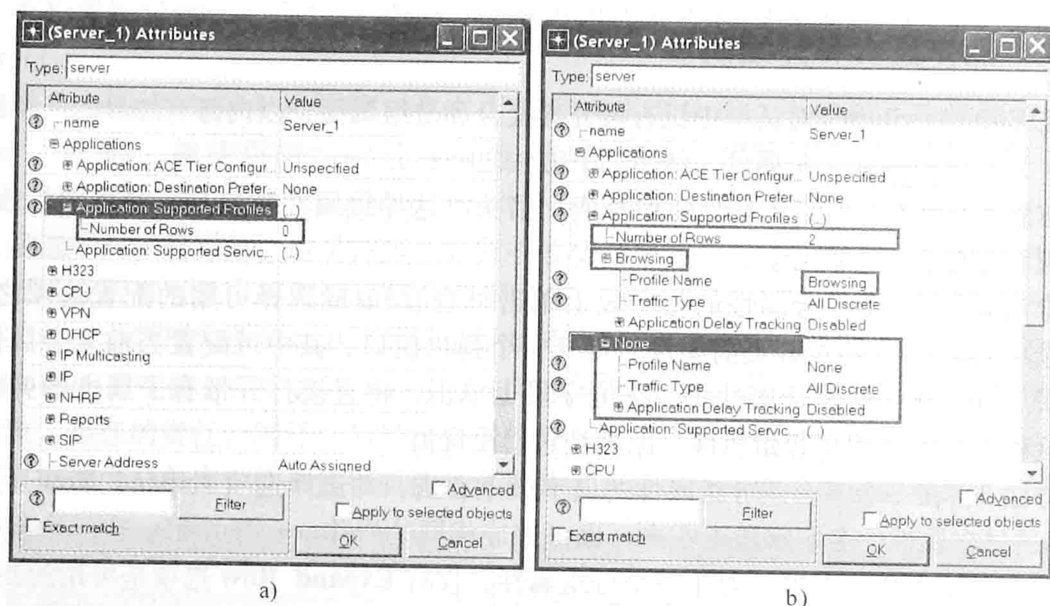


图 3.4　具有多个实例的属性

a）初始时，**Number of Rows** 为 0

b）当 **Number of Rows** 改为 2

3.3.4　过滤属性

Edit Attributes 对话框也基于选择准则，提供了过滤属性的一项功能。当使用 **Edit Attributes** 对话框的常规版本和高级版本时，过滤功能是不同的。利用常规的 **Edit Attributes**，仅可将过滤准则指定为一个子字符串，该字符串将与属性的名字、值或一个标签进行匹配。仅有具有搜索准则的部分匹配或完全匹配的属性才在 **Edit Attributes** 对话框中是可见的。

高级 **Edit Attributes** 对话框过滤功能使您可指定如何定义匹配准则（例如，准确的字符串匹配、部分字符串匹配或正则表达式）和应该搜索哪些属性性质进行匹配（例如，属性名、值、可能值、标签或上述字段的任意组合）。回顾一下第 1 章，其中标签是与模型的属性关联的"技术性的"关键字。OPNET 文档建议使用协议名、接口名或属性值作为属性的最高效过滤的搜索准则。图 3.5 给出常规和高级 **Edit Attributes** 对话框中过滤功能的外观。

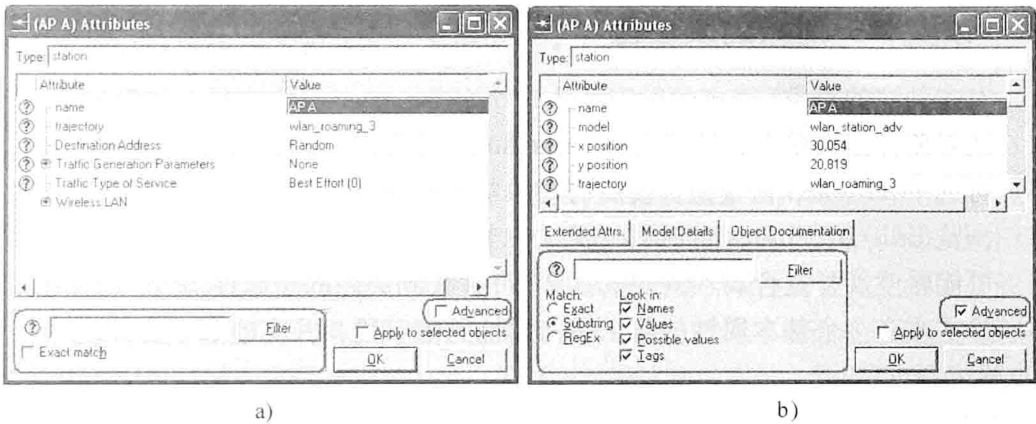
3.3.5　使用常规 Edit Attributes 过滤功能来寻找属性

1）在关注的对象上右击，并选择 **Edit Attributes** 选项，打开 **Edit Attributes** 对话框。

2）在 **Filter**（过滤器）按钮旁边的文本框中输入搜索准则。

3）单击 **Filter** 按钮，应用过滤准则。

4）为了去除过滤选择，清空文本框，并再次单击 **Filter** 按钮。



a) b)

图 3.5 **Edit Attributes** 对话框中的过滤选项

a）常规的 b）高级的

5）为了解有关过滤功能的更多信息，在位于过滤器搜索准则文本框左侧、内部有一个问号的圆圈形式的图标上单击。

3.3.6 使用高级 Edit Attributes 过滤功能来寻找属性

1）右击关注的对象，并选择 **Edit Attributes**（**Advanced**）选项，打开 **Edit Attributes** 对话框的高级视图。

2）如果 **Edit Attributes** 对话框处在常规视图，那么在 **Advanced** 检查框上单击，切换到高级视图。

3）如果需要的话，通过改变 *Match* 单选按钮的如下选项之一：exact（准确的）、substring（子字符串）或 RegEx（正则表达式），改变匹配准则。

4）如果需要的话，通过在标题 *Look in*（详细查看）下选择或取消选择检查框，如 Names（名字）、Values（值）、Possible Values（可能值）或 Tags（标签），改变要搜索的属性性质的类型。

5）在 **Filter** 按钮旁边的文本框中输入搜索准则。

6）在 **Filter** 按钮上单击，应用过滤准则。

7）为了去除过滤选择，清空文本框，并再次单击 **Filter** 按钮。

8）为了解有关过滤功能的更多信息，在位于过滤器搜索准则文本框左侧、内部有一个问号的圆圈形式的图标上单击。

3.4 配置对象性质

配置一个对象的最常见方法是改变它的属性值。本节描述如何改变单一对象的属性值以及同时改变多个对象属性值的方便的方法。

3.4.1 改变单一对象的属性值

1）右击关注的对象。

2）从对象弹出菜单中，选择 **Edit Attributes** 或 **Edit Attributes**（**Advanced**）。

3）可选中 *Advanced* 检查框，使所有对象的属性是可见的。

4）浏览 **Edit Attributes** 对话框（见 3.3 节），定位关注的属性。

5）可能希望展开复合属性（见 3.3.2 节），找到关注的子属性。

6）为了改变一个基本属性的值，右击属性的值字段，并实施如下操作之一：

① 输入新的值。

② 从提供的下拉菜单中选择一个值。

③ 从下拉菜单中选择 Edit... 选项，并输入一个新值。

7）一旦所有的属性值都被改变，单击 **OK** 按钮，关闭窗口并保存改变。

3.4.2 改变多个对象的属性值

经常的情况是，有必要改变多个对象中的同一个属性值。考虑这样一种场景，其中希望仿真具有 50 台工作站的一个网络拓扑，其中每台工作站都运行 TCP 的 Reno 风格版。明显的是，以个体方式（逐个地）配置 50 台工作站将花费很长时间。幸运的是，OPNET 提供了同时配置多个对象的几种方法。

1）选择要修改的对象。所有这些对象不必使用同样的模型，但它们应该至少有一个共同的属性。为了在网络拓扑中选择对象，实施如下动作之一：

方法 1：在项目工作空间上单击，之后在关注对象所在地方的面积（area）上拖动鼠标。

方法 2：在按住 Ctrl 键时，单击希望选择的对象。

方法 3：右击要选择的对象之一，之后从 **Object Pop - up Menu**（对象弹出菜单）（见 3.2.2 节）中选择选项 **Select Similar Nodes**（选择类似节点）。这个操作将选择在当前场景中具有同一模型的所有对象。应该指出的是，即使位于网络层次结构的不同层次（如子网）上但具有同一模型名的对象，作为这项操作的结果也被选择。

方法 4：在顶层下拉菜单中选择 **Edit→Select All in Subnet**（编辑→选择子网中的所有对象）。这将选择在当前子网中的所有对象。

方法 5：在顶层下拉菜单中选择 **Edit > Select Objects...**（编辑 > 选择对象）。这将打开一个 **Define Selection**（定义选择）窗口，这将使您可依据某个准则（如一个属性的对象类型或可用性）在整个场景（包括所有子网）中寻找和选择节点。

2）在选中所有期望的对象后，现在右击任何被选中的节点，并选择 **Edit Attributes** 选项。

3）选中 *Apply Changes to Selected Objects*（将改变应用到被选中的对象）检查框，从而在这个节点上实施过的改变将传播到在前一步中被选中的所有节点。这是一个重要步骤，当处于匆忙中时经常会忘记。忘记选中这个检查框，将导致改变仅发生在前一步

右击的一个节点，而不是在所有其他预选中的对象上。

4）实施对象性质的必要改变，遵循在 3.4.1 节中给出的步骤。

5）当完成时，单击 **OK** 按钮。在这个节点上实施的改变将被应用到所有被选中的节点。

OPNET 会提示您确认是否希望将改变实施到所有被选中的对象上。

就编辑多个对象的属性而言，存在一些要小心谨慎处理的附加提示点。如果选择了多个对象且改变了一个复合属性的任意子属性的值，那么整个复合属性将被复制到所有被选中的对象。这可能导致不期望的效果。同样，如果 OPNET 确定被请求的改变不能应用到一个或多个被选中的对象，那么将没有改变发生；但是，对于这样的失败，将得不到通知。

3.4.3　编辑被选中的对象

OPNET 提供了改变对象的属性值的另外一种方法，方法是在网络中编辑被选中的对象：

1）选择属性要被修改的所有期望的对象，使用的是在 3.4.2 节的第一步骤中描述的 5 种方法中的任意一种方法。这些对象不必具有同一模型。

2）右击项目工作空间，并从弹出菜单中选择 **Edit Selected Objects**（编辑被选中的对象）选项。出现的 **Objects Attributes**（对象属性）对话框，包含具有每个被选中对象一行的一个表。表中的列包含在所有被选中对象中存在的所有属性。注意具有相同名字的属性出现在相同列。也给出了每个对象的各属性的当前值。如果一个特定对象没有某个属性，那么其值就不会显示，相反出现一个空（蓝色）的框。

3）可将这个表中任意对象的属性值改变为任何期望的值。

4）在更新所有的必要属性之后，单击 **OK** 按钮，关闭窗口并保存改变。

3.4.4　编辑类似节点或链路

我们描述的最后一种方法可用来仅改变类似对象的属性。这种方法仅对节点和链路对象是可用的。

1）右击关注的对象之一（它必须是一个节点或一条链路），且从对象弹出菜单中，依据实际情形，选择 **Edit Similar Nodes**（编辑类似节点）或 **Edit Similar Links**（编辑类似链路）。

2）这个动作也打开 **Objects Attributes** 对话框，但它将仅包含类似对象，即具有同一模型的对象。在当前场景中具有同一模型的所有对象都将被显示。

3）将属性的值改变为期望的值，并单击 **OK** 按钮，保存改变。

3.4.5　一个对象的模型属性

通过一个称为 **model**（模型）的高级属性，定义最重要的对象性质，它定义了在当前对象中正被建模的网元。OPNET 运行这个高级属性的值可被改变。但是，当改变 **model** 属性值时一定要极其小心谨慎，原因是这样的一个改变可能导致被创建网络系统

中的不一致。例如，考虑这样一个场景，其中一个正确配置的网络拓扑，*Email_ client* 节点，将其 **model** 属性从以太网工作站（*ethernet_ wkstn*）改变为点到点协议工作站（*ppp_ wkstn*）。这样的一种修改将导致一个不正确的系统配置，原因是 *ethernet_ wkstn* 节点模型要求一条以太网链路，而 *ppp_ wkstn* 仅支持一个 SLIP 接口。

另外，在某些情形中，改变一个对象的模型可能是极有帮助的。如果您还记得，OPNET 对象经常具有各个层次的派生（derivation）。具有 *int* 修饰符或根本没有修饰符的模型可能将某些属性隐藏了，而具有 *adv* 修饰符的模型会将所有模型属性对用户开放。因此，将一个对象的模型从具有 *int* 修饰符或根本没有修饰符改变为具有相同模型名但有 *adv* 修饰符，会提供对附加属性的访问，并允许用户更准确地配置被仿真的系统。

如下步骤被用来改变一个对象的 **model** 属性：

1）右击关注的对象，并选择 **Edit Attributes**（**Advanced**）选项，打开 **Edit Attributes** 对话框的高级视图。

2）如果 **Edit Attributes** 对话框处在常规视图中，那么单击 *Advanced* 检查框切换到高级视图。

3）在称为 **model** 的属性（从表顶部算起的第二个属性）的值字段上单击。

4）OPNET 会提供一条警告消息，提示您，改变模型属性会使一些属性无效或不一致。如果决定改变 **model** 属性，就单击 **Yes** 按钮。

5）将出现可用模型的一个滚动列表。滚动列表，并为当前对象选择一个新模型。

6）如果关注的模型没有出现在所提供的列表中，选择 Edit... 选项。

① 选择 Edit... 选项，就打开非常类似于 **Object Palette**（对象调色板）窗口（见 2.3 节）的一个窗口，并包含 OPNET 中存在的所有模型。

② 浏览模型列表，并为当前对象选择一个新的模型。

③ 单击 **OK** 按钮，关闭窗口。

7）单击 **OK** 按钮，接受改变，并关闭 **Edit Attributes** 对话框。

8）强烈建议，在改变 **model** 属性之后，针对一致性而验证网络配置（见 2.6.2 节，了解如何验证链路）。

3.5 提升对象属性

OPNET 软件依赖于一个模型层次结构，它包含几层抽象。层次结构的每层都有与之关联的一个特定编辑器。OPNET 模型层次结构的底层使用 *process models*（进程模型）来指定各种协议和网络技术。进程模型的实现使用了称为 *Proto - C* 的 C + + 编程语言的一个变种，并带有扩展的有限状态机转换图。**Process Editor**（进程编辑器）提供了开发并编译进程模型以及经常与之关联在一起的外部代码文件的功能。因此，OPNET 模型层次结构的最低层，称为 *process model level*（进程模型层），允许采用 **Process Editor** 对个体网络协议和技术进行建模。

在 OPNET 模型层次结构的下一层，称为 *node level*（节点层），使用 **Node Editor**（节点编辑器）定义个体网络设备。在节点层，一个网络设备被建模为一个或多个模块的集合，它们是通过报文流或统计线连接的，这为模块间通信提供了方法。一般而言，每个模块有与之关联的一个进程模型，它定义了那个模块的操作。因此，**Node Editor** 提供了对 OPNET 模型层次结构节点层的访问，在这里可对个体网络设备建模。

Project Editor 属于 OPNET 模型层次结构的下一层次，称为网络层。在这个层次，可指定被仿真系统的拓扑，并配置那个系统的各个组件。**Project Editor** 使您可组织节点模型（节点模型是在前一层采用 **Node Editor** 创建的）为一个网络拓扑，并之后在所创建拓扑中改变对象的属性，以便准确地表示仿真研究中被建模的网络。

OPNET 模型层次结构的最后一层处理仿真研究或整个被仿真系统的配置。层次结构的这个层次，称为仿真层次，没有与之关联的一个独立编辑器。通常，**Project Editor** 为指定如下仿真配置参数而提供方法，这些参数如仿真时长、在仿真运行过程中要收集的统计量、随机数生成器的种子以及影响整个仿真的其他属性。除了 **Process Editor**、**Node Editor** 和 **Project Editor** 外，OPNET 还包含其他编辑工具，如 **Link Editor**、**Path Editor**（路径编辑器）、**Demand Editor**（需求编辑器）、**Packet Format Editor**（报文格式编辑器）、**ICI Editor**、**Distribution**（**PDF**）**Editor**［分布（PDF）编辑器］和 **Probe Editor**（探针编辑器）。但是，这些编辑器是不太频繁使用的，在本书中不做描述。图 3.6 形象地给出 OPNET 模型层次结构。

这些编辑器的可用性是直接取决于正在使用哪个 OPNET 产品的。具体而言，OPNET Modeler 集成了所有上述的编辑器，而 IT Guru 仅提供了对 **Project Editor**、**Probe Editor**、**Distribution**（**PDF**）**Editor** 以及其他一些编辑器的访问能力。IT Guru 没有提供使用 **Process Editor** 和 **Node Editor** 分别改变进程模型和创建定制网络设备的设施。尽管如此，多数对象属性是在模型层次结构的最低层处过程模型内定义和使用的。这些属性的值可直接在进程模型和节点层次进行设置。但是，使属性的值仅在模型层次结构的最低层是可修改的，或使模型属性对每个节点模型是预设的，将极大地限制 OPNET 产品的灵活性。出于这个原因，OPNET 提供了一种机制，使您可使用 **Project Editor** 在网络和仿真层次设置属性值。配置模型属性的过程使它们的值可在模型层次结构的较高层指定，这种做法称为 *promotion*（提升）。

属性值可提升到节点、到网络，甚至到仿真层次。属性也可从子网络提升到其父型子网络（上一级子网络）。提升到仿真层次的属性被看作整个仿真场景的属性，因此要在仿真运行时进行设置。将属性提升到仿真层次，是一项非常方便的功能，因为这允许完成如下操作：

1）在单一位置改变常用的参数，而不是在网络拓扑中的各个对象上单击改变参数。

2）以不同的属性值，设置自动化的仿真运行。

3）迭代使用可能属性值的一个范围，而不是复制场景和改变某些属性的值。

4）在多个对象上聚集提升后的属性，这非常类似于同时设置多个对象的属性值。

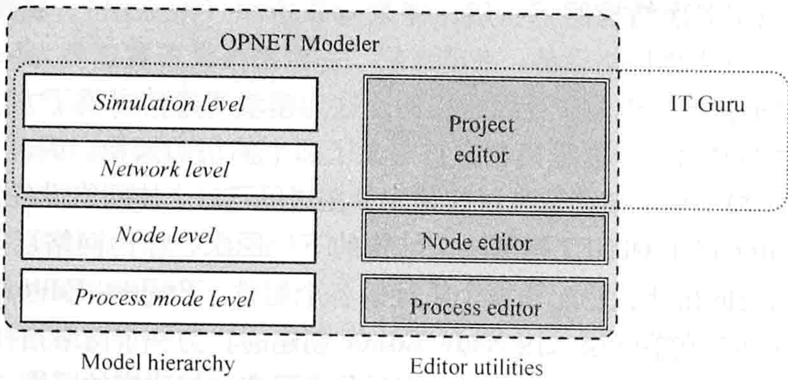


图 3.6 OPNET 模型层次结构

OPNET 不允许对没有与之关联值的属性进行提升，因此，分组属性是不能提升的。另外，高级属性和属性 **name** 不能提升。图 3.7 形象地给出属性提升操作。

被提升的属性值可在仿真层次或在父型子网层次进行设置[⊖]。一般而言，当一个属性被提升到模型层次结构的不同层次时，其名字依据如下命名惯例发生变化：

network_ type. subnet01. subnet02… subnet0n. node_ name. attribute_ name

例如，考虑一个被仿真的 *Campus Network*（园区网络），它包含带有子型子网 *sub01* 的子网 *sub_ top*。同样假定具有属性 **TCP Parameters** 的节点 *Admin01* 位于子网 *sub01* 内。表 3.1 给出当属性 **TCP Parameters** 被提升到模型层次结构的不同层次时，其名字如何改变。

表 3.1 属性命名惯例范例

模型层次结构的层次	属性的名字
Admin01	*TCP Parameters*
sub01	*Admin01. TCP Parameters*
sub_ top	*sub01. Admin01. TCP Parameters*
仿真层次	*Campus Network. sub_ top. sub01. Admin01. TCP Parameters*

在仿真层次，被提升的属性可被配置一个或多个值，或甚至具有以区间的形式指定的值。另外，OPNET 提供一种通配符特征，这使您可设置对网络中多个对象常见的被提升属性的值。本节后面部分为提升和取消提升属性、在模型层次结构的各层次配置被提升的属性值以及使用通配符表示等，提供了操作指令。

3.5.1 提升一个对象属性

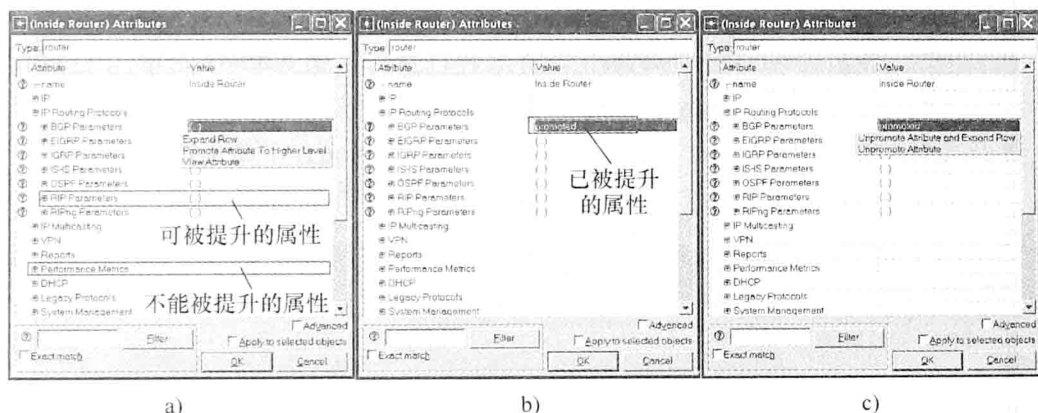
1）在关注的对象上右击，并选择 **Edit Attributes** 选项。

⊖ 在 OPNET 13.5 版和 15.0 版中，直到被提升的属性通过仿真范围的对象属性功能访问或查看时，它们可在父型子网层次可见。

2）在关注的属性的值字段上右击，并选择 **Promote Attribute To Higher Level** 选项（见图 3.7a）。属性的值将改变为 promoted（见图 3.7b）。

3）可依据需要在对象内提供足够多的属性。

4）当完成时单击 **OK** 按钮。



a)　　b)　　c)

图 3.7　属性提升

a）如何将一个属性提升到下一个较高的层次　b）已经被提升的一个属性　c）如何将一个属性取消提升

3.5.2　对一个对象属性实施取消提升操作

1）在关注对象上右击，并选择 **Edit Attributes** 选项。

2）有几种方式可以将一个属性值取消提升操作：

方法 1：右击取消所关注提升属性的值字段，并选择 **Unpromote Attribute**（取消提升属性）或 **Unpromote Attribute and Expand Row**（取消提升属性和展开行）（仅对复合属性是可用的）（见图 3.7c）。

方法 2：单击被提升属性的值字段，这可为属性设置一个值，自动地导致它取消提升。

3）依据需要可在对象内将足够多的属性取消提升。

4）当完成时单击 **OK** 按钮。

3.5.3　在仿真层次配置提升的对象属性

1）单击 *Run* 图标，或按 Ctrl + Shift + R 键，打开 **Configure/Run DES** 窗口。

2）当 **Configure/Run DES** 窗口出现时，可能需要切换到详细视图，方法是单击 **Detailed...** 按钮或按 Alt + D 键。

3）当详细视图 **Configure/Run DES** 窗口出现时，展开出现在左侧面中树视图中的 **Inputs** 控制页，之后单击 **Object Attributes** 选项。*Object Attributes Table* 现在显示在右侧面中。图 3.8 形象地展示了这个步骤。在 OPNET 的一些比较陈旧版本中，如 **AODV Hello Efficiency**（AODV Hello 效率）可能不会出现在如图 3.8 所示的 *Object Attributes Table* 中。单击 **Update**（更新）按钮就将表更新了，仅显示存在的和添加到 *Object At-*

tributes Table 中的那些属性。

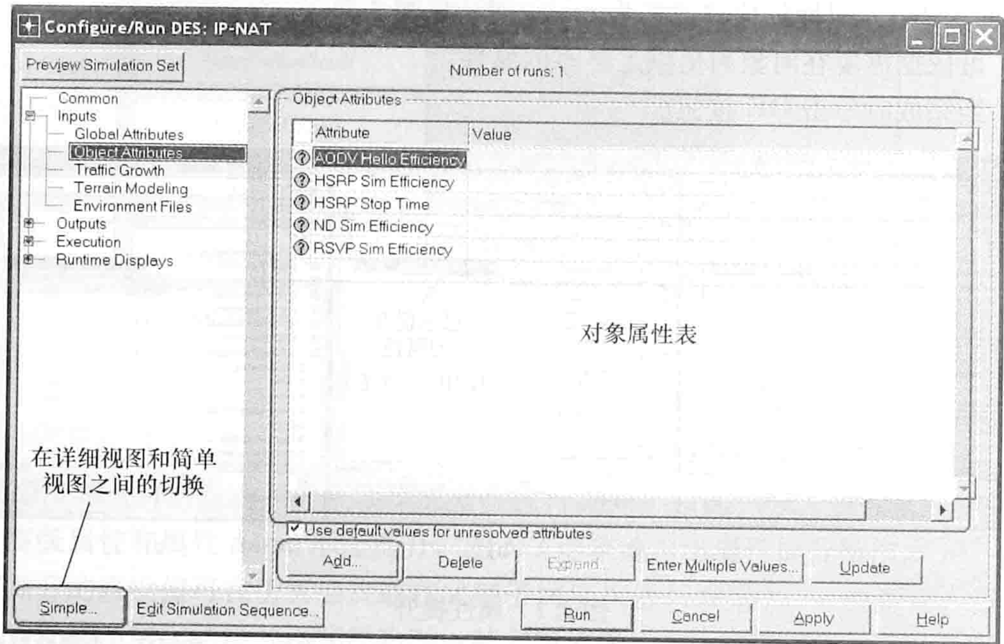


图 3.8 **Configure/Run DES** 窗口的 *Object Attributes Table*（对象属性表）

4）添加提升的属性：

① 单击 **Add** 按钮，打开 **Add Attributes** 窗口，这可将提升的属性添加到 *Object Attributes Table*。

② **Add Attributes** 窗口列出已经提升到仿真层次的所有属性。

③ 单击 *Add?* 列中的字段，标志要添加到 *Object Attributes Table* 中的属性。

④ 单击 **OK** 按钮，将标记过的属性添加到 *Object Attributes Table* 中。

3.5.4 在仿真层次为提升的属性指定值

一旦提升的属性已经放置到 *Object Attributes Table* 中，则可为它们指定值。

1）为 *Object Attributes Table* 中一个提升的属性指定单个值，与通过 **Edit Attributes** 对话框指定一个属性值，是没有区别的：单击 *Value* 字段，给出可用的预设值列表，在两次单击时，两次连续的单击或选择 Edit... 选项，可指定一个定制值。当编辑一个复合属性时，会被提供一个独立的窗口来指定子属性值。

2）为单一属性指定多个值：

① 单击属性名，选择它。

② 单击 **Enter Multiple Values**（输入多个值）按钮，这将弹出另一个窗口设置被选中属性的值。

③ 多个值可采取如下方法显式设置，每行指定一个值，或对于数值，在 *Value* 列指定初始值，在 *Limit* 列指定最大值，在 *Step* 列指定增量步。

④ 重复这些动作，在其他属性中设置多个值。

⑤ 单击 **OK** 按钮，接受选择的属性值，并关闭设置多个属性值的窗口。

图 3.9 形象地给出属性 **TCP Parameters [0]. Receive Buffer** 是如何被配置具有多个值的。在如图 3.9 所示的情形中，为这个属性指定了总共 17 个值。前三个值是以每行一个值的方式提供的，而其他 14 个值是以如下方式指定的，即从 70 000 到 200 000（包括在内）的区间，增量步长为 10 000。同样注意按钮 **Details**，在图 3.9 中以圆圈形式标出来了。单击 **Details** 按钮，就打开了包含正被配置属性的描述的一个窗口。

3）一旦所有被提升的属性的值都被设置，单击 **Apply** 按钮保存改变，或单击 **Run** 按钮执行仿真。如果单击 **Apply** 按钮，**Configure/Run DES** 窗口将保持打开状态。为了关闭 **Configure/Run DES** 窗口，单击 **Cancel** 按钮，或单击在窗口右上角的 *Close* 图标。

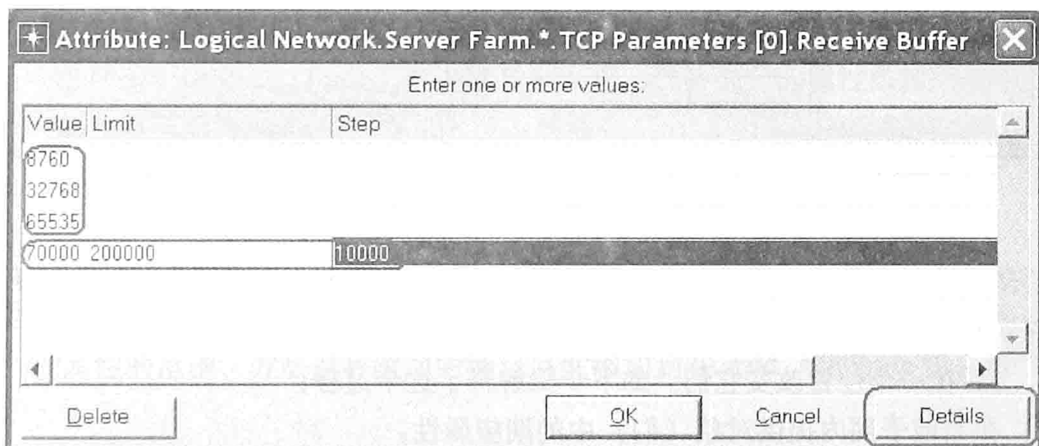


图 3.9　设置多个属性值

图 3.10 给出被提升属性的最终配置。窗口顶部给出为给定属性配置收集数据所要求仿真运行次数。这个值是 34，因为属性 **TCP Parameters [0]. Receive Buffer** 有 17 个不同值，且属性 **NAT Parameters** 有两个值，各值是以逗号分隔的；第一个值是 Not Configured，第二个值以（...）表示。因此，仿真运行总次数对应于选择属性值不同集合的可能组合数（17 × 2 = 34）。同样要注意称作 *Use Default values for unresolved attributes*（为未解析的属性使用默认值）的一个检查框，它位于窗口的底部部分，正好在 **Add...** 按钮上。选择这个检查框，将强制所有被提升的属性其值还没有设置的，都有默认值指派给它们。这个选项是非常有用的，原因是它确保所有属性都设置值。在不为一些属性指派值的情况下，运行仿真可能导致不可预测的且可能是不正确的结果。

注意：在 OPNET 软件的较陈旧版本中，建议选择称为 *Save vector and environment files for each run in set*（为集合中的每次运行保存向量和环境文件）的检查框。选择这个检查框，就确保为每次仿真运行都保存结果。从 OPNET 13.5 版开始，所有仿真运行的结果都自动保存，且在 **Configure/Run DES** 窗口中就不存在这个检查框了。

3.5.5　在子网层次配置提升的属性

提升在一个子网内对象的属性，之后当处在父型子网中时设置它们的值，这种做法

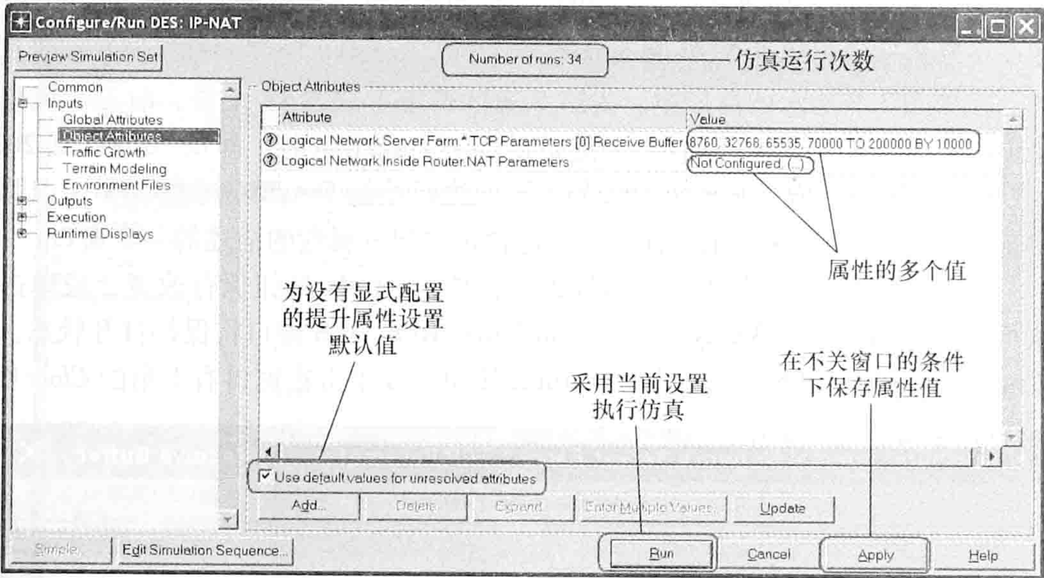


图 3.10 被提升属性的最终配置

当需要多次重新配置在子网中对象的属性时，是极其有用的。在这项功能的帮助下，与向下深入到子网，之后一次改变一个对象的属性值的做法不同，现在可依据需要，提升属性，之后在一个位置改变它们。如下步骤解释了这个过程：

1）在当前子网内提供对象（们）中的期望属性。

2）转到父型子网［在项目工作空间中右击，并选择 **Go To Parent Subnet**（转到父型子网)］。

3）在子网图标上右击，并选择 **Edit Attributes** 选项。

4）出现的 **Edit Attributes** 对话框，将包含属于位于被选择子网内各对象的所有提升属性的一个列表。

5）编辑属性值，并单击 **OK** 按钮，保存改变并关闭窗口。

3.5.6 使用通配符选项将值指派到多个提升的属性

考虑这样一个例子，其值被仿真的 *Logical Network*（逻辑网络）包含一个子网 *Server Farm*，其中有三个节点，名为 *Admin01*、*Admin02* 和 *Admin03*。所有这些节点都将属性 **TCP Parameters** 提升到仿真层次。结果，如下三个属性就存在于 **Configure/Run DES** 窗口中的 *Object Attributes Table* 之中：

1）**Logical Network. Server Farm. Admin01. TCP Parameters**

2）**Logical Network. Server Farm. Admin02. TCP Parameters**

3）**Logical Network. Server Farm. Admin03. TCP Parameters**

这些被提升的属性的名字是相同的，但带有每个属性所属的节点名。为了简化这些属性的配置，可利用通配符 * 表示属性名的共同部分的不同节点名：

Logical Network. Server Farm. *. TCP Parameters

因此，与在 **Configure/Run DES** 窗口的 *Object Attributes Table* 中列出每个对象一个属性的做法不同，可聚合具有相同名字但属于不同对象的被提升的属性。这样一种聚合的结果是使用通配符的一个属性的单一列表。但是，这种通配符聚合仅适用于被提升的属性。具有相同名字但没有被提升的属性，不会受到通配符聚合的影响。将一个值指派到通配符聚合属性，会导致具有那个名字的所有被提升属性都有这个相同的值指派给它们。也可选择缩小被聚合属性的范围，方法是将属性名中的通配符替换为一个具体的对象或子网名。

如下步骤描述如何使用通配符在仿真层次配置被提升的属性：

1）打开 **Configure/Run DES** 窗口，并选择 **Object Attributes** 选项。

2）在 *Object Attributes Table*，在 **Add** 按钮上单击，将提升的属性添加到表中。

3）在 **Add Attribute** 窗口，单击关注的属性。

4）如果被选中的属性是在通配符的帮助下指定的，那么按钮 **Expand** 将成为可见的。

① 单击 **Expand** 按钮，打开 **Expand Wildcard**（展开通配符）窗口。

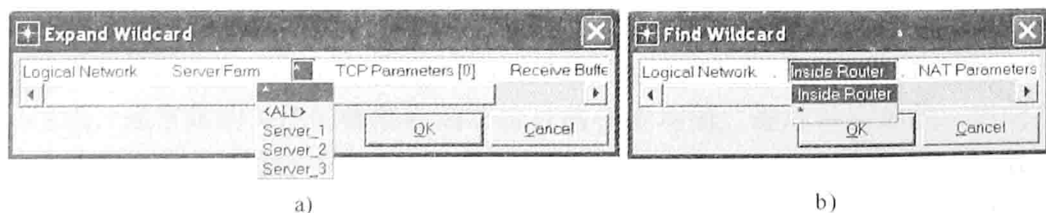
② 单击通配符，从出现的下拉菜单中为希望配置的属性选择一个具体对象。这将缩小当前属性的范围，方法是将附加的特定属性添加到在 **Add Attribute** 窗口中显示的列表中。

③ 单击 **OK** 按钮，保存改变，并返回到 **Add Attribute** 窗口。

5）如果在没有使用通配符的情况下，指定被选中的属性，那么按钮 **Wildcard...** 将成为可用的。

① 单击 **Wildcard...** 按钮，打开 **Find Wildcard** 窗口。

② 单击希望应用通配符的属性名部分，从下拉菜单中选择通配符。这个动作将展开当前属性的范围，包括满足新的通配符属性名的所有被提升的属性。



a)　　　　b)

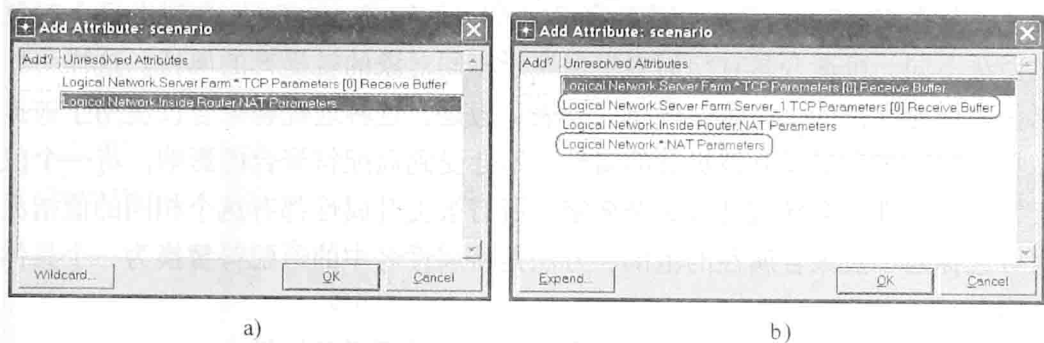
图 3.11　**Expand Wildcard** 窗口和 **Find Wildcard** 窗口

6）如果仅存在一个具有指定名字的提升属性，那么这个动作没有效果，否则一个新属性将被添加到 **Add Attribute** 窗口中的列表。

7）单击 **OK** 按钮，保存改变，并返回到 **Add Attribute** 窗口。

图 3.11 给出 **Expand Wildcard** 和 **Find Wildcard** 窗口，而图 3.12 给出在 **Receive Buffer** 属性中通配符为 *Server_1* 节点替换以及在 **NAT Parameters** 属性中 *Inside Router*

为通配符替换前后，**Add Attribute** 窗口看起来的样子。



a) b)

图 3.12 **Add Attribute** 窗口

a）在实施通配符操作之前 b）在 **Receive Buffer** 属性中通配符替换为 *Server_ 1* 节点，**NAT Parameters** 属性中的 *Inside Router* 替换为通配符之后

第4章 配置和运行一个仿真

开发一项仿真研究通常涉及创建要被仿真的系统的一个模型、选择仿真统计、配置仿真属性、运行仿真并研究收集到的结果。本书的第1~3章已经描述了使用OPNET软件如何开发被仿真系统的基本模型。在本章，将介绍如何运行仿真，包括在运行仿真之前如何配置仿真，并在完成运行之后查看和分析结果。

在执行一个仿真之前，必须选择在仿真运行过程中希望观察的统计数据。4.1节介绍OPNET中的仿真统计，4.2节描述为一项仿真研究选择和配置统计。在选择统计之后，存在必须配置的额外仿真属性。范例有仿真应该运行的时间量，或在仿真过程中要产生的随机数使用的初始种子。4.3节解释如何配置这些属性。4.3节也描述下一步，即如何执行仿真。在仿真完成运行之后，将关注于观察结果。4.4~4.7节描述如何显示和分析收集到的仿真结果。在一些情况中，仿真执行会是不成功的，原因是在构造模型或配置仿真中存在错误。4.8节描述使用离散事件仿真（DES）日志如何完成简单的调试，该日志记录在仿真运行过程中记录产生的某些重要消息。

4.1 OPNET中的仿真统计

一个仿真**statistic**（统计量）是一个或多个值的集合，描述仿真过程中进程行为的某些方面。仿真统计有助于发现促动仿真研究的问题的答案。它们提供了在仿真运行过程中被仿真系统中发生情况的深入理解。

在OPNET中，仿真统计通常作为一个结果文件存储在本地磁盘上。OPNET仿真进程收集统计值和值被记录时的时间，并将这些成对存储。每个值描述在发生某个事件或在一个特定时刻时的进程状态。OPNET将这样的统计称为**vector**（向量）数据。向量数据的结果文件是值和时间对的一个列表或向量。也可能的情况是，仿真收集的统计由单个值组成，通常是所关注的收集得到的统计的样本均值、最后得到的值、时间平均、方差以及最小值或最大值。OPNET称这样的统计为**scalar**（标量）数据。一般而言，收集结果的内容（可以是向量或标量数据）取决于针对收集的一个特定统计量，仿真是如何配置的。

例如，考虑一个统计量，它记录在某个节点中由IP模块发送的总流量。取决于统计的配置，在结果文件中存储的值可以是如下的任一种情况：

1）单一值，对应于IP发送的总流量。

2）值的集合，包含在特定时刻或发生特定事件时（如一条报文被调度离开IP）发送的IP流量总量。

3）值的集合，每个值对应于发送的IP流量总量，是在某个时间段（如每秒）上

平均得到的。

在 OPNET 中可进行收集的统计量分为两类：

1）对象统计量，描述在被仿真系统的一个特定对象内的进程行为。取决于对象是一个节点、一条链路或一项需求，这样的一项统计被称作一个 **Node Statistic**（节点统计量）、一个 **Link Statistic**（链路统计量）或一个 **Demand Statistic**（需求统计量）。

2）总体统计量，描述在整个被仿真系统中特定协议的行为。一个总体统计量由仿真中的所有对象（是那个特定类型的）共享，且所有这些对象对统计的总值都有贡献。

一个节点统计量（或一个节点的对象统计量）的例子是 **Client Ftp. Traffic Sent（bytes/sec）**［客户端 Ftp. 发送的流量（字节/秒）］。这项统计量测量一个特定节点的客户端 FTP 应用以字节/秒为单位发送的流量速率。如果统计量是在多个节点收集的，那么将针对每个节点单独地测量和收集流量。一个全局统计量的例子是 **Ftp. Traffic Sent（bytes/sec）**［Ftp. 发送的流量（字节/秒）］。这项统计测量在被仿真网络中在所有节点处 FTP 应用以字节/秒为单位发送的总流量速率（所有的都加到单一量上）。

4.1.1 统计收集模式

一个统计量的 **collection mode**（收集模式）指定统计值将被收集的方式。知道一个统计的收集模式，对于理解为那个统计报告的值的意义，是非常重要的，且它会严重地影响对仿真研究尝试解决问题的答案。一个统计量可被收集为一个向量数据或一个标量数据。当收集为 **vector data**（向量数据）时，在 OPNET 中存在三种主要的统计收集（或捕获）模式：

1）**all values**（所有值）——为发生的每个相关事件，记录所有的统计值。

2）**sample**（样本）——仅收集所有统计值的一个样本。采样的方式可以是计数法（如每隔 5 个数据电记录一次）或时间区间法（如每 6 秒记录一次数据）。

3）**bucket**（桶）——在以时间区间或计数（如 10 秒或 50 个值）指定的一个桶上观察到的所有值，并为每个桶产生单一结果值（在桶上所有值的和或平均）。对于许多常用的统计量，这是默认的收集模式。

当以 **all values**（所有值）模式收集时，为影响关注的统计的每个事件记录统计值。因此，这种模式提供了在仿真研究中发生情况的最准确描述。但是，所有值模式得到巨量的收集数据，这经常是难以解释的，因此，这些结果通常情况下在理解仿真过程中发生事件方面是没有用处的。

例如，在一个节点的外发接口上一条 IP 报文的离去，是直接影响 **IP. Traffic Sent（packets/sec）**［IP. 发送的流量（报文数/秒）］统计的事件。结果文件将包含带有时间—值对的一个列表的向量数据，其中每对都记录观察到的每个事件的事件时间和统计值。如果以所有值模式记录统计，则对每个离开 IP 层的报文收集其值。因为统计值是发送的报文数，所以对每个事件这个值是 1。结果是时间—值对的一个长列表，其中对于每个对，时间是一条报文的离开时间，值为 1。这不仅得到收集巨量数据，而且该数据是难以解释和理解的。例如，如果数据是以 1000 个报文数/秒的平均速率传输的，那

么对于以所有值模式收集的 **IP. Traffic Sent**（**packets/sec**）统计，10min 的仿真将得到大约 600 000 个数据点（时间—值对）。

第二种收集模式，**sample**（样本）模式，以时间区间（如每 10 秒）或以计数（如每第 10 个事件）就收集一个样本值。这种技术大量地降低了收集到的数据量。但是，采样必须小心地进行设计，以便确保它代表了要被监测的性质。如果要监测链路利用率，且在链路上的报文流没有一个确定性的模式（如每 5s 一次报文传输），那么以时间区间采样就将工作得很好。另外，在每第 5 次报文传输上的链路利用率采样，将不能提供代表性的值，原因是对每个样本，利用率都将为 1。依据计数采样的方法，比较适合于离散量，如队列长度，其中可得到每第 6 个到达或从队列离开的队列长度的一个样本。

第三种收集模式，**bucket**（桶）模式，是收集仿真统计量的最常用方法。在桶模式中，类似于样本模式，统计值是在定义为时间区间或以计数方式的时段上收集的。为每个时段记录的仅有一个值，称为一个 **bucket**。但是，桶模式取一个桶中统计的所有观察值，并处理它们形成单一值，之后记录该值。处理可产生所有值的和，或其平均值，或它可使用可应用于桶中所有值的某个其他函数。输出向量的尺寸正比于桶数：每个桶一个值，这恰像在样本模式中一样。同样，类似于样本模式，必须小心使用桶模式，以避免做出不正确的结论。

当一个统计量被作为 scalar data（向量数据）收集时，得到整个仿真的单一值。通过应用求和、平均、概率、最小、最大或其他统计函数，通常可从所有观察的统计值中得到一个标量统计值。作为标量数据收集的统计例子包括一个节点发送的报文总数、平均队列长度、最小分段延迟等。就其本身而言，一个标量值提供处理行为的非常狭小的视图。出于这个原因，当收集标量值时，同一仿真通常以系统的配置参数值（会发生改变）之一多次执行。之后，这些多次仿真运行的结果被收集并以值对的形式加以记录：<*value of configuration parameter that was changed*>（被改变的配置参数值）和 <*the resulting scalar value of the statistic*>（得到的统计标量值）。例如，考虑这样一项仿真研究，尝试详细研究传输速率对最大队列占用率的影响。在这样一项研究中，仿真被执行多次，最大队列占用率收集为一个标量统计量，此时传输速率值是变化的。结果向量将包含值对的一个集合：针对每次仿真运行要有一对。每对将包含最大队列占用率的记录值和在那次仿真运行中配置的相应传输速率。这样一项研究的结果可画为一幅图：最大队列占用率对传输速率。

4.1.2　确定要收集哪些统计量

OPNET 提供多样化种类的仿真统计量以及配置其收集模式的能力。但是，OPNET 不会自动地选择一次仿真过程中要收集的统计量。默认情况下，一次仿真根本不收集统计量。即使不为收集选择统计量，仿真仍然执行。这样一种仿真不是非常有用的，原因是它没有提供被建模的系统中发生了什么的信息。确定哪些统计量与研究相关并应该在仿真运行过程中收集，这是用户的职责。

因此，当设计一项仿真研究时，必须确定在仿真运行中应该选择哪些统计量。清楚的是，配置仿真来收集所有可用的统计量，并不是一个良好的思路。这样一个配置将显著地降低仿真执行速度，并将消耗大量计算资源。相反，仿真应该被配置来仅收集对仿真研究所必要的那些统计量。

但应该收集哪些统计量呢？最重要的且最明显的选择是选择与仿真模型的配置直接相关的统计量。例如，如果仿真已经配置运行一个 FTP 应用，那么要谨慎选择 FTP 统计量。类似地，如果仿真研究显式地配置某些协议和技术，那么描述这些以及其他密切相关的协议和技术行为的统计量，将是收集的良好候选。通常情况下，基于某些常见特点（如协议），组合个体统计量。但是，即使为一次仿真研究显式配置一个协议，收集特定协议的所有可用统计量，不是一个良好思路。

下一项选择是同样明显的，并处理可有助于回答仿真研究问题的统计量。例如，如果仿真的目标是详细研究一个 FTP 应用对网络层性能的影响，那么即使仿真仅利用网络层协议的默认配置值，也应该收集合适的网络层统计量。

最后，选择并不直接与仿真研究相关，但也可提供对被仿真网络运行的深入理解的那些统计量，经常是一个不错的想法。这样的统计量可有助于调试和验证仿真模型。例如，收集 Link Statistics（链路统计量）（如吞吐量和利用率），这经常是有用的，这种做法有助于确定流量如何通过网络链路以及由个体节点产生多少流量。网络中的流量模式可帮助识别没有被正确配置的节点。另外，链路利用率可解释报告得到的端到端响应时间（例如，当链路利用率接近 100% 时，流量延迟会指数增加）。另外，不应该收集在仿真研究中不用的统计量。因此，如果电子邮件或 UDP 这两种协议在被研究网络中不用的话，则没有理由收集这两类统计量。

总之，确定要在一个仿真中被收集的统计量，是一个迭代过程。当仿真仍然正在被开发且要求调试的话，则收集比必要的要多的统计量，从而可更好地理解被仿真系统的行为是明智的。一旦多数配置问题被修正且仿真展示出稳定行为，则一些统计量就不再需要，并可从配置中被去除以便加速仿真执行。最终的仿真可仅收集有助于形象说明研究的要点的那些统计量。

4.2 选择仿真统计量

4.2.1 选择结果窗口

OPNET GUI 为在仿真过程中选择要收集的统计量提供一个 **Choose Results**（选择结果）窗口。打开 **Choose Results**（选择结果）有三种方式：

1）右击任意对象（节点或链路），并从弹出菜单中选择 **Choose Individual DES Statistics**（选择个体 DES 统计量）。

2）在工作空间中的任意地方右击，之后从弹出菜单中选择 **Choose Individual DES Statistics**（选择个体 DES 统计量）。

3）在 Project Editor（项目编辑器）中进入 DES 菜单，并选择 **Choose Individual DES Statistics...**（选择个体 DES 统计量...）。

Choose Results（选择结果）窗口的外观和可用于选择的统计量类型，取决于用于打开它的这三种方法而稍稍有所不同。回顾一下 4.1 节，统计量可以是全局的或对象统计量，其中全局统计量收集被仿真系统中一个特定进程的行为有关的信息（作为一个整体），而对象统计量收集单一对象中有关进程行为的信息。对象统计量进一步分类为节点、链路或需求统计量，这取决于关注的对象分别是一个节点、一条链路或一个需求对象。

当通过在一个对象上右击一个对象而打开 **Choose Results**（选择结果）窗口时，仅有那个对象的统计量才可被选择。因此，通过在网络中一个特定节点上右击，选择节点统计量 **IP. Traffic Dropped**（**packets/sec**）[IP. 丢弃的流量（报文数/秒）]，将报告在那个节点的 IP 层中每秒丢弃的报文数。当通过在项目工作空间中右击或通过 **DES** 菜单，打开 **Choose Results**（选择结果）窗口时，可选择全局统计量和对象统计量，但是，对象统计量将为网络拓扑中与统计量相关的所有对象而进行个体性的收集。例如，如果选择全局统计量 **IP. Traffic Dropped**（**packets/sec**）[IP. 丢弃的流量（报文数/秒）]，那么它将报告被仿真网络中所有节点中每秒丢弃的报文总数。另外，如果通过在项目工作空间中右击，选择节点统计量 **IP. Traffic Dropped**（**packets/sec**）[IP. 丢弃的流量（报文数/秒）]，那么为网络中的所有节点收集这个统计量，但各个值可个体性地用于每个节点，而不是像在全局统计量中的情形那样被加总。出于这个原因，以这种方式收集的对象统计量有时也称为 **Scenario - wide Statistics**（场景范围的统计量）。

不管 **Choose Results**（选择结果）窗口是如何打开的，则在所有情形中选择特定统计量的进一步指令是相同的。图 4.1a 形象地展示了具有场景范围的统计量的 **Choose Results**（选择结果）窗口。如图 4.1a 所示，**Choose Results**（选择结果）窗口包括全局、节点和链路统计类，并进一步分为子类，子类基于常见协议、技术和应用等将统计量归组到一起。也可能的情况是，**Choose Results** 窗口包含其他统计类，如需求统计或模块统计。仅当仿真被配置包括某些特征功能时，这些统计类才可用于收集。例如，仅当被仿真网络模型包括需求对象时，才可收集需求统计。除了在窗口中显示类的数量和类型外，处理统计类的指令是相同的。为了简化讨论，仅描述全局、节点和链路统计类，这是仿真研究中最常用的。全局和节点统计经常具有非常类似的子类，通常依据协议类型进行聚集（如 IP、TCP、BGP 和电子邮件）。链路统计仅包括两个子类：*low - level point - to - point*（低层点到点）和 *point - to - point*（点到点），收集有关物理链路数据传输的信息。

描述 **Choose Results** 窗口中统计量结构组织的另一种方式是作为一棵三层树组织。顶层包括全局、节点和链路通用类，进一步基于技术或网络协议分为子类。最终的树层包含实际的统计量。如图 4.1a 所示，上两层中的表项包含加号或减号形式的图标，支持展开或收缩类视图（即单击加号，展开类视图，并将图标改变为包含一个减号）。

在加号或减号的图标旁边，每层包含一个选择检查框，指明是指定类中的统计量还

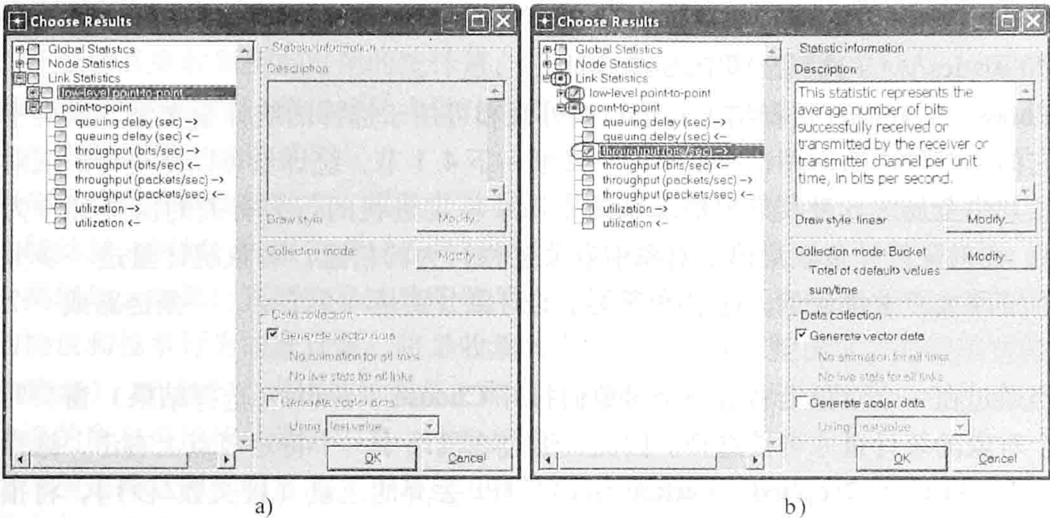


图 4.1　**Choose Results**（选择结果）窗口

a）没有选择统计量　b）选择一些统计量

是一个个体统计量被选中进行收集。这个检查框有三种可能的设置：

1）*Clear*（清除）指明在类内的所有统计量或一个个体统计量在仿真过程中不被选择。在如图 4.1b 显示的窗口中，在全局和节点类中不选择统计量，且链路类中许多个体统计量（如 **utilization - >**（利用率 - >））也不选择。

2）*Marked with a green checkmark*（以一个绿色的检查框标记）指明为仿真过程中的收集选择一个个体统计量还是类中的所有统计量。在图 4.1b 中，在仿真过程中将收集链路统计量 **throughput（bits/sec）- >**（吞吐量（位/秒））和 *low - level point - to - point*（低层点到点）类中的所有统计量。

3）*Marked with a green dot*（以一个绿色点标记）指明给定类或子类中的一些（而不是所有的）统计量被选择进行收集。绿色点指示法不适用于个体统计量。如图 4.1b 所示，仅有 *point - to - point*（点到点）和链路统计类中的一些统计量被选择进行收集。

在一个个体统计量上单击检查框，可交替选中/取消那个统计量的选择状态。在一个二层类的检查框上单击，可选择或取消那个类中的所有统计量。OPNET 不允许通过单击相应检查框而支持顶层类（全局、节点或链路）中的所有统计量的选择。为了选择顶层类中一个类内的所有统计量，必须分别选择所有的相应子类。

4.2.2　为单一特定网络对象选择仿真统计量

1）在位于 **Project Editor**（项目编辑器）内所关注的对象上右击。

2）从弹出菜单中，选择 **Choose Individual DES Statistics**（选择个体 DES 统计量），将打开一个 **Choose Results** 窗口（带有可用于被选对象的统计量）。

3）通过单击加号或减号，展开或收缩统计类，直到定位关注的统计量（可能是多项），采取这种方式浏览 **Choose Results** 窗口内的统计树。展开统计量将使加号改变为减号。

4）单击一个个体统计量的检查框，选择它。重复这个过程，直到选择关注的所有统计量。为了选择一个子类中的所有统计量，单击相应的检查框。

5）单击 **OK** 按钮，关闭窗口，并保存改变。仅对被选对象，才收集所选择的统计量（可能是多个）。

4.2.3　为整个场景选择仿真统计量

1）在 **Project Editor**（项目编辑器）的工作空间内任意地方右击。从弹出菜单中选择 **Choose Individual DES Statistics**（选择个体 DES 统计量），打开 **Choose Results** 窗口。另外一种方法是，进入 **Project Editor** 的 **DES** 菜单，并选择 **Choose Individual Statistics...**。

2）在 **Choose Results**（选择结果）窗口内浏览统计树，并选择期望的统计量（可能有多个）。这些步骤与4.2.2节中的那些步骤是相同的，例外情况是现在可浏览节点统计类和/或链路统计类。

3）单击 **OK** 按钮，关闭窗口并保存改变。

上述过程将使选中的统计量（可能是多项）在所有适用的网络对象中单独进行收集。

4.2.4　选择全局仿真统计量

1）在 **Project Editor** 的工作空间内任意地方右击。从弹出菜单中选择 **Choose Individual DES Statistics**，这打开一个 **Choose Results** 窗口。另外一种方法是，进入 **Project Editor** 的 **DES** 菜单中，并选择 **Choose Individual Statistics...**。

2）在 **Choose Results** 窗口内浏览统计树，并选择期望的统计量（可能有多项）。这些步骤与4.2.2节和4.2.3节描述的那些步骤是相同的，例外情况是现在可浏览全局统计类。

3）单击 **OK** 按钮，关闭窗口并保存改变。

4.2.5　统计信息和数据收集面

Choose Results 窗口的右侧部分包含统计配置参数和有关被选择统计量的一些附加信息。如图4.2所示，窗口的这部分包含两个面，标记为 *Statistic information*（统计信息）和 *Data collection*（数据收集）。

如图4.2所示，*Statistic information* 面包括被选统计量的一个简短描述，并显示了统计的图示风格和收集模式，以及改变这些设置的选项。**statistic draw style**（统计的画线风格）控制当显示结果时，统计将如何在一幅图中画出。位于 *Draw style* 文本框右侧的 **Modify**（修改）按钮将使您可改变统计量的画线风格。在画线风格一节下面显示 **statistic collection mode**（统计收集模式），并控制如何收集统计值。位于 *Collection mode*（收集模式）文本框的右侧，有一个 **Modify**（修改）按钮，这使您可改变统计的收集模式。

最后，如图 4.2 所示，**Choose Results** 窗口包含一个 *Data collection*（数据收集）面，它直接位于 *Statistic information*（统计信息）面。这个面有两个检查框：*Generate vector data*（产生向量数据）和 *Generate scalar data*（产生标量数据）。可选择检查框中的任何一个或两者都选。如果选中 *Generate scalar data* 选项，那么也就有必要指定将如何计算标量值。在这种情形中，可为计算标量值从如下函数中进行选择：last value（上一个值）、sample mean（样本均值）、time average（时间平均）、variance（方差）、min value（最小值）或 max value（最大值）。

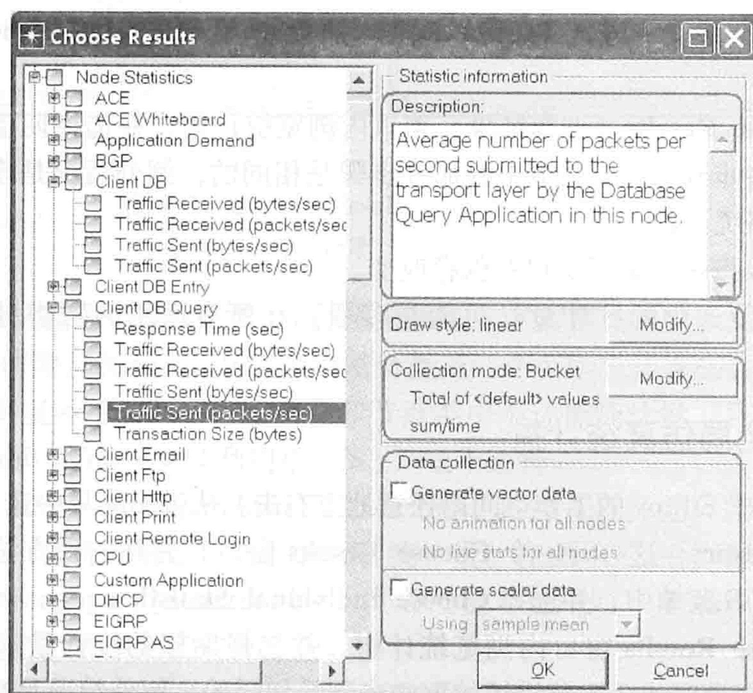


图 4.2 **Choose Results** 窗口的 *statistic information* 和 *Data collection* 选项面

4.2.6 统计画线风格

单击在 Draw style（画线风格）文本框旁边的 **Modify**（修改）按钮，打开一个窗口，如图 4.3 所示，选择一个统计画线风格。在窗口中列出的期望画线风格上单击，并单击 **Close**（关闭）按钮，保存修改。被选中的画线风格将被用于显示针对这项统计以一种图形形式而收集的数据。

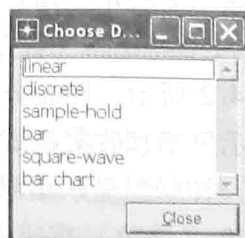


图 4.3 统计项的 *Draw Style* 选项

可用的画线风格如下：

1）Linear（线性）——当在一幅图形中显示时，数据点以直线线段连接起来。

2）Discrete（离散）——数据点显示为离散点，没有线连接它们。

3）Sample – hold（样本保持）——将每个点扩展为一条水平线，直到显示下一个样本之前，这本质上维持（或保持）样本值。

4）Bar（柱形）——样本保持风格的一种扩展，其中为每个样本所画的水平线一直被填充到水平轴。

5）Square – wave（方波）——就像在样本保持风格中一样，为每个点画水平线，但之后以到下一个样本点的垂直线连接它们。实际上，这是一种长条风格，但却没有填充。

6）Bar chart（柱形图）——一种传统的柱形图形，其中从每个数据点到水平轴画一条垂直线。

4.2.7　统计收集模式

配置 **Statistic Collection Mode**（统计收集模式）的选项位于画线风格节下面。在位于 **Collection mode**（收集模式）文本框右侧的 **Modify**（修改）按钮上单击，打开一个 **Statistic Collection Mode** 窗口，可改变统计项的收集模式。默认情况下，**Statistic Collection Mode** 窗口以常规视图打开，这仅支持一些基本收集模式性质的修改。单击 *Advanced*（高级）检查框，将窗口切换到一个高级视图，这就允许对其他性质的修改。图 4.4a 和图 4.4b 分别给出 **Statistic Collection Mode** 窗口的常规视图和高级视图。

Capture mode（捕获模式）选项仅存在于高级视图中，且这使您可配置数据以何种方式被收集。可选择如下捕获模式：

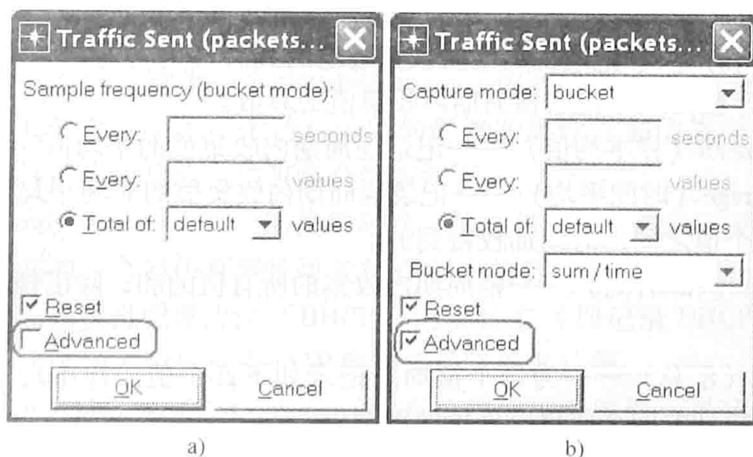


图 4.4　**Statistic Collection Mode** 窗口
a）常规视图　b）高级视图

1）*All values*（所有值）——收集所有数据点。

2）*Sample*（样本）——仅记录来自所有可用数据集合的样本数据点。通过一个时间区间（例如，每 2 秒收集数据值）或一个样本计数（例如，每第 10 个数据值时收集），指定样本数。没被采样的数据点被忽略。

3）*Bucket*（桶）——默认的收集模式，将在称为一个桶的时段内发生的数据点归

组，之后将一个统计函数应用到值的每个组。得到的输出向量为每个桶包含一个值。与样本模式一样，时段可被指定为一个时间区间或一个样本计数。

4）*Glitch removal*（去除瑕疵）——如果在同一时刻记录多个统计值，那么这种统计模式去除所有重复的数据点，仅保留记录的最后一个值。

接下来的三个单选按钮仅适用于桶和样本收集模式。在窗口的常规视图和高级视图中，这些单选按钮都是可用的。在高级视图中，如果选择所有值或瑕疵去除模式，那么这些按钮都变灰，指明这些按钮对那些模式是不可用的。同样，在那种情形中，不能切换回到常规视图。这些单选按钮分别指定了值有多频繁地针对桶模式和样本模式进行聚集或采样：

1）每 ***N*** 秒——在每 ***N*** 秒后将记录一个值。

2）每 ***N*** 个值——仅每第 ***N*** 个值被记录。

3）总共 ***N*** 个值——输出向量将包含总共 ***N*** 个值对，这意味着每个（*duration of simulation/N*（仿真时长/*N*））时间单位记录一个值。在这种情形中，***N*** 被称作每个统计量的值数（*values per statistic*）。其默认值是 100，通过编辑首选项 **num_ collect_ values**（见 1.2.1 节）或在 **Config/Run DES** 窗口（见 4.3.1 节）中改变相应属性值的方法，可改变这个值。

bucket mode（桶模式）文本框（仅存在于高级视图中）右侧的下拉菜单，仅适用于桶收集模式。它指定桶收集模式功能，可以是如下情况之一：

1）max value（最大值）——记录桶周期内收集的最大值。

2）min value（最小值）——记录桶周期内收集的最小值。

3）sum（求和）——记录桶周期内收集各值的和。

4）count（计数）——记录桶周期内收集值的总数。

5）sample mean（样本均值）——记录桶周期内收集值的平均值。

6）time average（时间平均）——记录桶周期内收集值的时间平均。时间平均是依据当前值和下一个值之间的时间加权得到的。

7）sum/time（和/时间）——桶周期内收集的所有值的和，除以桶周期时长，记录得到的结果。

8）summary（汇总）——为每个桶周期记录如下四个值：样本均值、最大值、最小值和标准差。得到的四个值集以单独的图画出。

Reset（重置）检查框（存在于常规视图和高级视图中）指定在计算下一个桶值之前，来自前一个周期的桶值是否重置为零。选择 *Reset* 检查框（是默认的），确保桶值仅依赖于在那个特定桶周期过程中收集的数据值。将 *Reset* 检查框取消选中操作，将在当前桶周期过程中计算值时结合以前桶值。在某些情形中，这可能是有帮助的。例如，如果桶模式用于 *max* 函数，那么选中 *Reset* 检查框，将产生每个桶内的最大值，但取消选中 *Reset* 检查框，将得到从仿真开始直到每个桶的时间点时产生的最大值。

4.2.8 修改统计收集性质

1）在 **Choose Results** 窗口中，选择关注的个体统计量。

2）在 *Statistic information*（统计信息）面中：

① 在位于 *Draw Style*（画线风格）标签右侧的 **Modify** 按钮上单击，改变在结果图中统计显示风格。

② 在位于 *Collection mode*（收集模式）标签右侧的 **Modify** 按钮上单击，改变统计值如何被收集的方法。

3）在 *Data collection*（数据收集）面中：

① 如果希望统计量以数据向量的形式进行收集，那么选中 *Generate vector data*（产生向量数据）检查框。

② 如果希望统计量作为单个值被报告，那么选中 *Generate scalar data*（产生标量数据）检查框。使 *Generate vector data* 和 *Generate scalar data* 检查框都不被选中，将导致对当前统计量不收集数据。

③ 在其他个体统计量上重复相同过程。

④ 单击 **OK** 按钮，关闭窗口并保存改变。

上述指令不可应用于统计类。一般而言，所有被选中的统计量将保持其默认配置，除非它们使用前面的指令进行显式修改。对于多数仿真研究而言，标准统计量的默认配置是足够的，且不需要改变。但是，如果引入一个新的统计量或如果仿真研究需要被收集结果的一个不同表示，那么就需要遵循上述指令，并依据需求改变统计配置。

4.3　配置和运行一个仿真

本节描述仿真属性［称为一个 **simulation set**（仿真集合）］和配置与运行一个仿真的步骤。一个仿真集合是描述一个仿真运行的配置的属性（如仿真时长）和提升到仿真层次的模型属性的一个集合。仿真集合组成如下：

1）*Global model attributes*（全局模型属性）描述被仿真系统而不是系统内一个特定对象的各方面。例如，全局仿真属性可包括在整个仿真网络中使用哪种 IP 路由协议的全局仿真属性，动态主机配置协议（DHCP）记录日志（即记录 DHCP 活动），IP 路由表输入选项（即包括输入到模型中的 IP 路由表的文件名）等。

2）*Simulation - wide model attributes*（仿真范围的模型属性）描述对多个对象通用的一个仿真模型的各方面。通常而言，这些是被提升到仿真层次的属性（见 3.5 节）。

3）*Simulation - run attributes*（仿真运行的属性）描述一个给定仿真系统的仿真运行的配置。仿真运行属性包括配置参数，如仿真时长、随机数生成种子、每个统计量的值数量、流量增长规格、动画、OPNET 调试器（ODB）的使用以及其他。

配置仿真集合和执行仿真通常是通过 **Configure/Run DES** 窗口或 **Simulation Sequence Editor**（仿真序列编辑器）完成的任务。但是，仿真场景也可通过 **Project Editor** 中的 **Scenarios→Manage Scenarios**（场景→管理场景）下拉菜单选项来运行，见 1.7.2 节的描述。为了打开 **Simulation Sequence Editor**，从下拉菜单中选择 **DES→Configure/Run Discrete Event Simulation（Advanced）**［DES→配置/运行离散事件仿真

(高级)] 选项。通过实施如下的任何一项动作，可打开 **Configure/Run DES** 窗口：

1）单击 *running man*（奔跑的人）图标（）。

2）按 Ctrl + R 键。

3）从下拉菜单中选择 **DES→Configure/Run Discrete Event Simulation...** 选项。

在这些仿真配置设施之间的主要差异是，**Configure/Run DES** 支持每个仿真模型的一个仿真集合的配置，而 **Simulation Sequence Editor** 是一个比较高级的选项，可被用来为同一仿真模型创建多个仿真集合。在多数情形中，对于实施一个简单的研究，单一仿真集合就足够。值得指出的是，仿真集合数未必等于仿真运行的次数。一个仿真集合是一个给定仿真的值的集合，而仿真运行的次数是给定仿真配置将执行的仿真的次数。一些仿真属性可被配置一个以上的值，这将导致同一仿真集合的多次执行。

4.3.1 Configure/Run DES 窗口：简单模式

配置一个仿真的最常见方式是采用 **Configure/Run DES** 窗口。这个创建包含两个视图模式：simple（简单的）和 detailed（详细的）。简单模式设置仅支持一些关键仿真参数的配置，而详细模式提供配置控制的一个完备集合，其中包括指定多个属性值的一个选项。如图 4.5 和图 4.6 所示，两种模式都包含一个按钮，该按钮支持从详细视图模式切换到简单视图模式，反之亦然。在简单模式中，在 **Configure/Run DES** 窗口底部的按钮 **Detailed...**（见图 4.5）切换到详细视图模式。类似地，在详细模式中，在窗口左下角的按钮 **Simple...** 将窗口切换到简单视图模式。

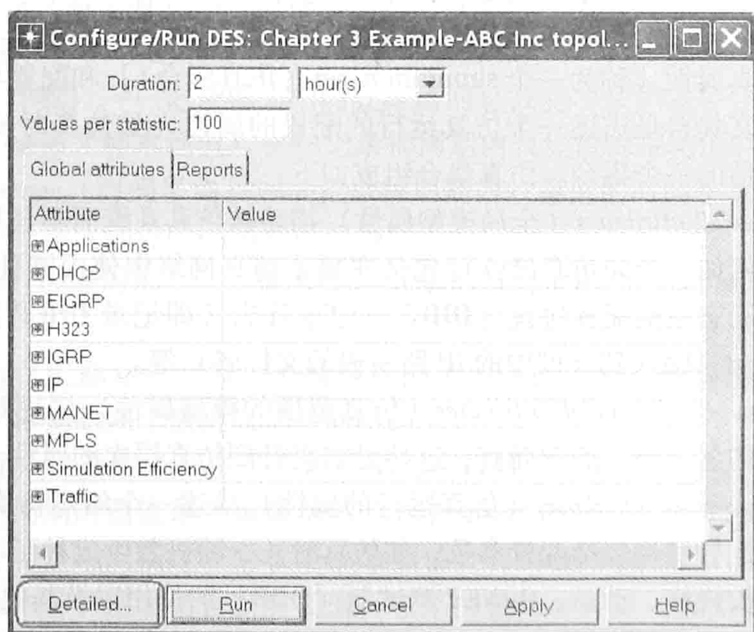


图 4.5 **Configure/Run DES** 窗口的简单视图

如图 4.5 所示，简单模式支持指定两个常用的仿真集合属性。如下：

1）*Duration*（时长）——仿真将运行多长时间或在被仿真的环境内被建模的系统

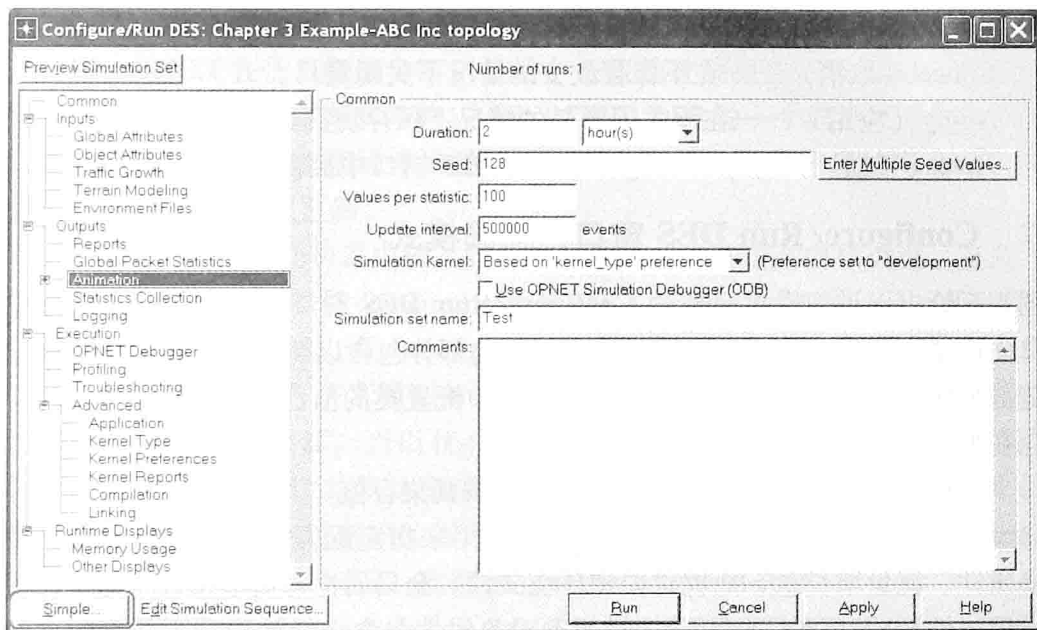


图 4.6　**Configure/Run DES** 窗口的详细视图

将处于可操作状态的时间总量。通常，仿真时长不同于完成仿真运行所花费的实际时间。例如，考虑这样一个网络模型，其仿真时长属性值被设置为 10h。如果网络模型是十分简单的，那么仿真运行将在数秒内完成。另外，如果被建模的系统包含运行复杂协议的大量节点，那么仿真运行的完成将花费长得多的时间，原因是一个复杂的模型将典型地产生大量生成的事件，处理要花费较长时间。

2）*Values per statistic*（每个统计量的值数）——对于每个被选中的统计量，报告的数据点的最大数量。如 4.2.7 节所述，这个属性被用来计算在桶中收集的统计量的桶长度和样本模式。如果所有仿真统计量都被配置为在所有值都收集或小瑕疵去除收集模式的话，则这个属性值被忽略。

Configure/Run DES 窗口的简单视图包含两个 tabs（标签），通过它们可指定两个额外的属性组。这些组如下：

1）*Global attributes*（全局属性）——在仿真中使用的各种协议的全局属性，在这里以一个树视图的方式是可访问的。每个协议名可被展开查看可用的属性。也存在一个子组称为 *Simulation Efficiency*（仿真效率），它包含针对各种路由协议控制一个仿真的效率的各属性（见 11.2.5 节和 11.3.11 节）。最后，子组 *Traffic*（流量）包含与背景流量和流量中各流有关的属性（见 6.6 节）。

2）*Reports*（报告）——这个组使您可选择统计报告，以便在一个网页浏览器中查看。每个统计报告是在仿真过程中收集的统计量的一个集合，但做了合适的处理，从而可在一个浏览器中查看。在本书中不再进一步讨论报告。

在 **Configure/Run DES** 窗口底部的按钮有如下含义：

1）**Detailed...** 详细——切换到 **Configure/Run DES** 窗口的详细视图。

2）**Run**（运行）——保存仿真集合各值，关闭窗口，并执行仿真。

3）**Cancel**（取消）——在不保存改变的情况下关闭窗口。

4）**Apply**（应用）——在不关闭窗口的情况下保存仿真集合各值。

5）**Help**（帮助）——在 **Configure/Run DES** 窗口中提供选项的一个描述。

4.3.2　Configure/Run DES 窗口：详细模式

图 4.6 给出以详细模式呈现的 **Configure/Run DES** 窗口的一个例子。详细视图模式窗口由两个不同节（section）组成。窗口左边部分包含以树形式组织的配置类，而窗口右侧部分包含一组控制，用于设置一个被选中配置类的值。

存在如下配置类：

1）*Common*（通用）类包含最频繁使用的仿真集合值。

2）*Inputs*（输入）类包含几个子类，可被用来指定配置项，如全局属性、被提升的对象属性、流量增长量、地貌建模和环境文件。全局属性与简单模式中这个标签下的那些可用属性相同（见 4.3.1 节）。被提升对象属性包含已经被提升到仿真层次的所有属性（见 3.5 节）；这里可提供这些属性的值。流量增长子类使您可指定在给定仿真模型中随时间推移流量总量的增长有多快/多慢。对于确定服务水平协议（SLA）的界限，这个子类是有用的。地貌建模要求额外的地貌建模许可证，因此这里不做讨论。一个环境文件是一个纯文本文件，它包含仿真集合配置选项和被提升属性的值。为了简化配置多个仿真集合的任务，创建几个环境文件，之后使用 *Environmental Files*（环境文件）子类指定要在当前仿真运行中使用的环境文件，是可能的。

3）*Outputs*（输出）类由几个子类组成，这允许配置在仿真完成时要创建报告、收集仿真统计的时间时长、仿真日志数据收集、动画等。多数这样的配置属性是自解释的。

4）*Execution*（执行）类处理仿真执行的各种 OPNET 设施的配置。例如，这个类支持配置 ODB 和轮廓生成器（profiler）、指定排错信息、设置仿真内核的细节（即并行还是串行，32 位还是 64 位地址空间等）、提供编译和连接选项等。

5）*Runtime*（运行时）显示类包含配置仿真进展报告的属性，这些报告是在仿真执行过程中在 **Simulation Progress**（仿真进展）窗口中显示的。

默认情况下，**Configure/Run DES** 窗口以详细视图模式打开，为配置选中 *Common*（通用）类。*Common* 类是最频繁使用的仿真配置视图之一，且它支持指定常用的配置属性：

1）*Duration*（时长）为仿真的时长（在 4.3.1 节描述）。

2）*Seed*（种子）为随机数生成器的种子值。位于种子文本框右侧的按钮使您可指定多个种子值。当希望通过采用种子值的一个不同集合重新运行同一仿真得到多个统计上准确的结果（即更精确的置信区间）时，这个特征是有用的。

3）*Values per statistic*（每个统计量的值数）为每个选中的统计量，报告的数据点的最大数量（在 4.3.1 节描述）。

4）*Update interval*（更新间隔）确定仿真进展报告在仿真控制台中显示的频率。当仿真执行时，OPNET 在仿真控制台窗口显示一条短的进展报告。*Update interval* 的值指定在连续进展报告之间的仿真事件数。一个 OPNET DES 的一个典型尺寸是数百万个事件。因此，*Update interval* 的值通常设置为几十万个事件，这在仿真的寿命期上得到仿真控制台的 10 次或更多次更新。

5）*Simulation Kernel*（仿真内核）指定用来执行仿真的仿真内核类型。有两种内核类型：development（开发）和 optimized（优化的）。**development kernel**（开发用内核）支持仿真的密切监测和调试，而 **optimized kernel**（优化的内核）为快速执行而优化了代码。开发用内核通常用作仿真模型的开发和调试，而优化的内核一般用作在仿真被验证过之后多次执行大型仿真。当以优化的内核执行一个仿真时，ODB 是不可用的。**Simulation Kernel** 配置属性的默认值依据的是 **kernel_ type** 首选项的值（见 1.2 节）。默认情况下，**kernel_ type** 首选项的值为 development。显式地将 **Simulation Kernel** 属性值设置为 development 或 optimized，就覆盖了 **kernel_ type** 首选项。

6）*Use OPNET Simulation Debugger*（*ODB*）[使用 OPNET 仿真调试器（ODB）] 是一个检查框，当选中时，使用 ODB 执行仿真。当 **Simulation Kernel** 属性值设置为 optimized 时，这个选项是被禁用的。

7）*Simulation set name*（仿真集合名）是当前仿真集合的一个字母数字组成的名字。这个属性的值对仿真执行没有影响。但是，当使用 **Simulation Sequence Editor**（仿真序列编辑器）创建多个仿真集合时，选择一个好的和描述性的仿真集合名是有益处的。

8）*Comments*（注释）是一个文本框，可用之指定有关当前仿真集合的注释或提示。这个属性对仿真运行也没有影响。

在我们的经验中，多数仿真仅要求设置仿真集合属性的值，如仿真时长、种子、每个统计量的各值、某些全局和提升的对象属性，且也许有被选中的报告。其他的仿真属性通常保持不变，并设置为其默认值。

Configure/Run DES 窗口的详细视图模式包含了一些其他按钮，它们在简单模式中是不可用的。下面进行描述：

1）**Simple...**（简单的）——将 **Configure/Run DES** 窗口切换到简单模式。

2）**Previewsimulation set**（预览仿真集合）——打开一个文本窗口，显示一条命令行语句（invocation），带有当前仿真集合执行所关联的程序参数。这个窗口也显示环境文件的内容和全局与被提升的对象属性非默认值的列表。这个信息可作为编写脚本在 OPNET GUI 外部运行仿真的一个例子。

3）**Edit Simulation Sequence**（编辑仿真序列）——打开 **Simulation Sequence Editor**（仿真序列编辑器）。

4.3.3 仿真序列编辑器

Simulation Sequence Editor 是一个 OPNET 工具，可被用来指定和操作（如复制、

粘贴、剪切、删除、选择或执行）多个仿真集合。每个仿真集合是由其名字和使用这个集合执行的仿真运行次数确定的。在仿真集合的名字上右击，之后选择 **Edit Attributes** 选项，就打开带有仿真集合一个详细视图的 **Configure/Run DES** 窗口。如图 4.7 所示，**Simulation Sequence Editor** 有三个仿真特定的快捷图标：

1）*blue arrow*（蓝色箭头）图标（ ），它创建另一个仿真集合。

2）*running man*（奔跑的人）图标（ ），它执行检查框被选中的所有仿真集合。

3）*log book*（日志簿记）图标（ ），它打开仿真日志。

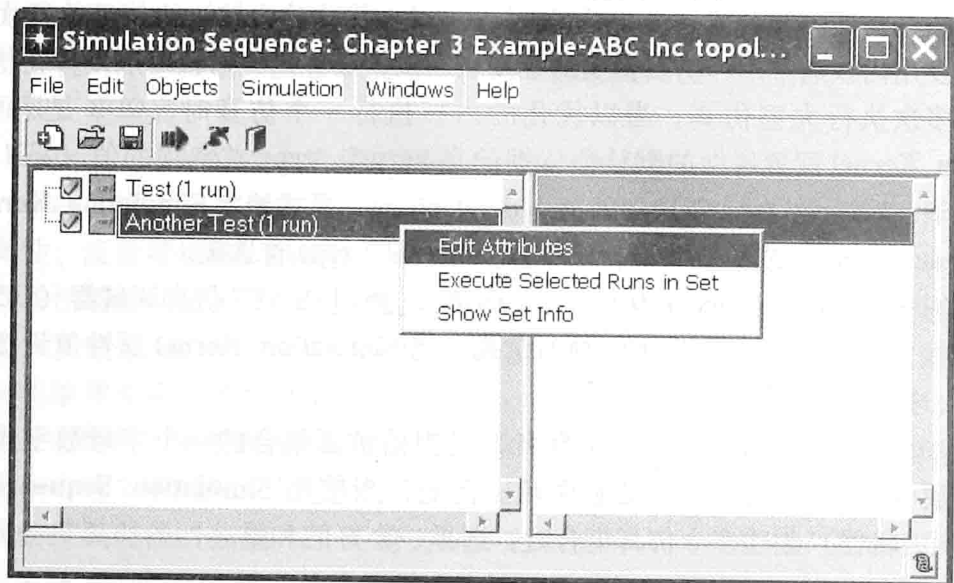


图 4.7 **Simulation Sequence Editor**

通过 **Simulation Sequence Editor** 的下拉菜单，所有这些选项也都是可用的。

作为 **Simulation Sequence Editor** 的替代方法，也可通过在 **Project Editor** 中可用的 **Manage Scenarios**（管理场景）选项执行仿真集合（见 1.7.2 节）。回顾一下在 4.3.5 节中通过 **Manage Scenarios** 选项执行仿真的指令。最后，OPNET 也支持从命令提示来进行仿真执行。但是，因为这超出了本书的范围，所以不讨论这个特征。

4.3.4 配置和执行单个仿真场景

1）从 Project Editor 内，单击 *running man*（奔跑的人）图标（ ），按 Ctrl + Shift + R 键，或从下拉菜单中选择 **DES→Configure/Run Discrete Event Simulation...**。

2）当 **Configure/Run DES** 窗口出现时，也许需要切换到详细的视图，方法是单击 **Detailed...** 按钮或按 Alt + D 键。

3）在 **Configure/Run DES** 窗口中，如果需要，修改仿真集合的属性。

4）单击 **Run** 按钮，执行仿真。

4.3.5 通过管理场景来配置和执行多个仿真场景

1）在 **Project Editor** 中，从下拉菜单中选择 **Scenarios→Manage Scenarios**。

2）在出现的 **Manage Scenarios** 窗口中，将所关注场景的 *Results*（结果）栏标记设置为 < collect >（收集）或 < recollect >（重新收集）。

3）在已经选择所有场景［希望（重新）收集结果的］之后，单击 **OK** 按钮。这将使所有被选中的场景顺序运行。执行或重新执行一个场景将重写任何以前收集的结果。

4.3.6　为提升的属性设置值

有时，来自模型层次结构低层次的属性可被提升到仿真层次，如 3.5 节所述。如果存在这样的提升属性，那么在仿真可被运行之前，必须为它们提供值。OPNET 可为单个被提升的对象属性指定多个值。如果那样做，那么仿真将被执行多次，对被提升属性的每个值执行一次。如果存在多个被提升的属性，则仿真运行的次数等于仿真集合中每个属性值数量的乘积。如果每个属性仅有一个值，那么仿真将仅运行一次。但是，如果有 k 个属性，每个属性具有 N_i 个值，那么运行的总次数由如下公式定义：

$$\text{仿真运行次数} = N_1 \times N_2 \times \cdots \times N_k = \prod_{i=1}^{k} N_i$$

仅有被提升到顶层子网的那些属性才可被配置多个值。参见 3.5 节，了解有关属性提升的更多细节。

使用如下步骤设置被提升属性的值：

1）从 **Project Editor** 内部，单击 *running man*（奔跑的人）（）图标，按 Ctrl + Shift + R 键，或从下拉菜单中选择 **DES→Configure/Run Discrete Event Simulation...**。

2）当 **Configure/Run DES** 窗口出现时，需要切换到详细的视图模式，方法是通过单击 **Detailed...** 按钮或按 Alt + D 键。

3）在 **Configure/Run DES** 窗口的详细视图中，展开出现在窗口左侧部分的 *Inputs*（输入）类，之后单击 *Object Attributes*（对象属性）子类。

4）为了添加被提升的属性：

① 单击 **Add** 按钮，打开 **Add Attributes**（添加属性）窗口，将被提升的属性添加到 *Object Attributes Table*（对象属性表）中。

② **Add Attributes**（添加属性）窗口列出已经被提升到顶层子网层次（即仿真层）的所有属性。

③ 单击 *Add?* 列中的字段，将属性标记为添加到 *Object Attributes Table*。

④ 为添加一个以上的属性，重复这些步骤。

⑤ 单击 **OK** 按钮，将做过标记的属性添加到 *Object Attributes Table*。

5）在属性已被放置在 *Object Attributes Table* 内时，就能够配置被提升属性的值。为一个被提升的属性指定单个属性值，与通过 **Edit Attributes** 对话框指定一个属性值没有什么不同：在 *Value*（值）字段单击，给出可用的预设值的一个列表，而双击键（两次连续的单击），或选择 Edit... 选项，可指定一个定制的值。当编辑一个复合属性时，可能提供指定子属性值的一个独立窗口。

6）为单个属性指定多个值：

① 单击属性名，选中它。

② 单击 **Enter Multiple Values**（输入多个值）按钮，这将弹出设置被选属性值的另一个窗口。

③ 通过显式地每行指定一个值，或对于数值型的值，通过在 *Value* 列指定一个初始值，在 *Limit* 列指定最大值，并在 *Step* 列指定递增步长，这样就可设置多个值。

④ 为其他属性设置多个值，请重复这些动作。

⑤ 单击 **OK** 按钮，接受选择的属性值，并关闭窗口。

7）一旦所有被提升的属性的值都设置了，那么单击 **Run** 按钮，执行仿真。如果至少有仿真设置的一个属性配置了多个值，那么在 **Configure/Run DES** 窗口顶部的文本标签将列出大于 1 的运行次数。

4.3.7　仿真执行

一旦单击 **Run** 按钮，仿真将开始执行，且一个称为 **Simulation Execution**（仿真执行）窗口的新窗口将出现。如果仿真是从 **Configure/Run DES** 窗口以简单视图模式开始的，那么 **Simulation Execution** 窗口也将以一种简单模式出现，如图 4.8 所示。**Simulation Execution** 窗口的简单模式由三个平板组成：

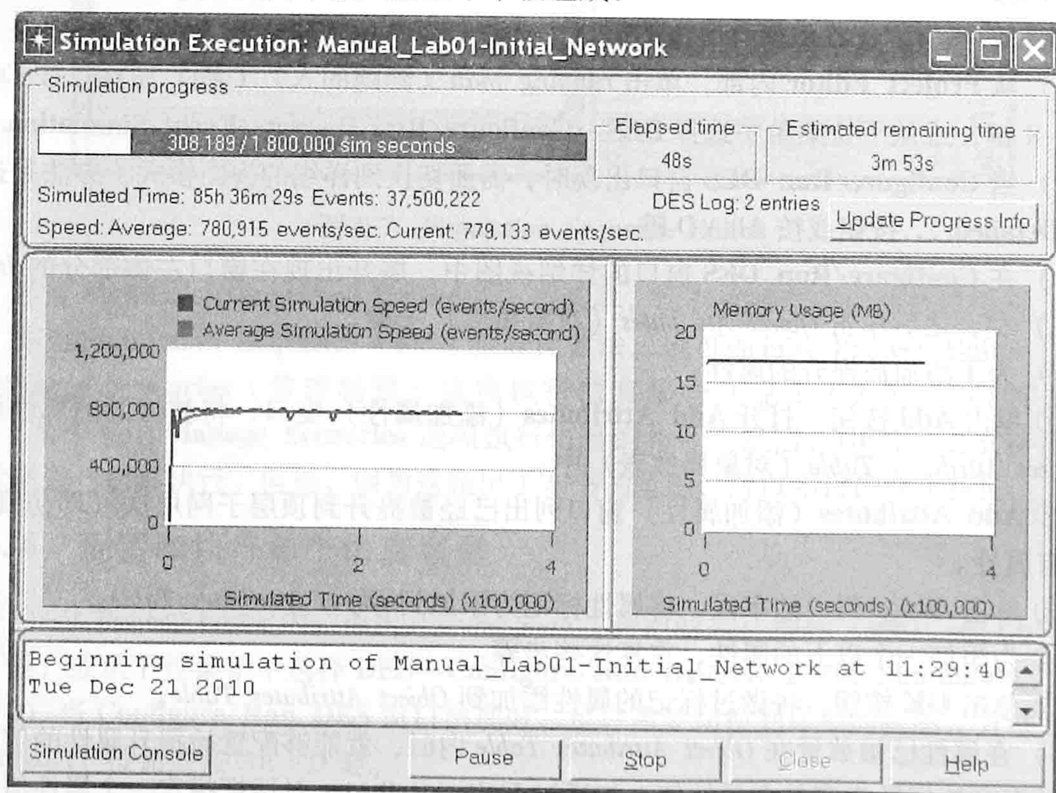


图 4.8　**Simulation Execution** 窗口的简单模式

1）*Simulation progress*（仿真进展）包含一个进展条，显示过去的仿真秒数与仿真总秒数的比值。这个平板也显示过去的仿真时间、到此为止产生的仿真事件总数、以事件/秒为单位表示的平均和当前仿真速度、自开始仿真以来消逝的真实时间、直到仿真

完成时的估计真实时间和到此为止产生的DES日志表项总数。*simulation progress* 平板也包含一个 **Update Progress Info**（更新进展信息）按钮，当单击时，立刻以最新数据更新仿真进展。

2）*simulation speed and memory usage*（仿真速度和内存使用情况）平板包含两个图形：一个图形显示实际的和平均的仿真速度，另一个图形给出在当前仿真运行过程中的内存消耗情况。

3）*messages*（消息）平板显示由仿真产生的消息（描述自动化仿真进展的那些消息除外）。

在 **Simulation Execution** 窗口底部有5个按钮：

1）**Simulation Console** 按钮打开一个独立的仿真控制台窗口，该窗口包含由仿真产生的所有消息，其中包括自动化进展更新和仿真初始化消息。

2）**Pause**（暂停）按钮将仿真执行挂起。当仿真挂起时，**Pause** 按钮改变为 **Resume**（恢复）按钮，当该按钮被按下时，恢复仿真执行。

3）**Stop**（停止）按钮终止仿真执行，导致仅有部分仿真结果被收集（即仅有仿真停止之前的结果被收集）。

4）**Close**（关闭）按钮关闭 **Simulation Execution** 窗口。在仿真执行过程中，该按钮是被禁止的，仅在仿真完成或停止之后才变为可用的。

5）**Help**（帮助）按钮打开一个帮助窗口，提供在 **Simulation Execution** 窗口中可用特征的一个简洁描述。

处于详细模式的 **Simulation Execution** 窗口，如图4.9所示，包含单个平板，有在 *Simulation progress* 平板之下几个选项卡（tabs），与详细视图模式和简单视图模式中的完全相同：

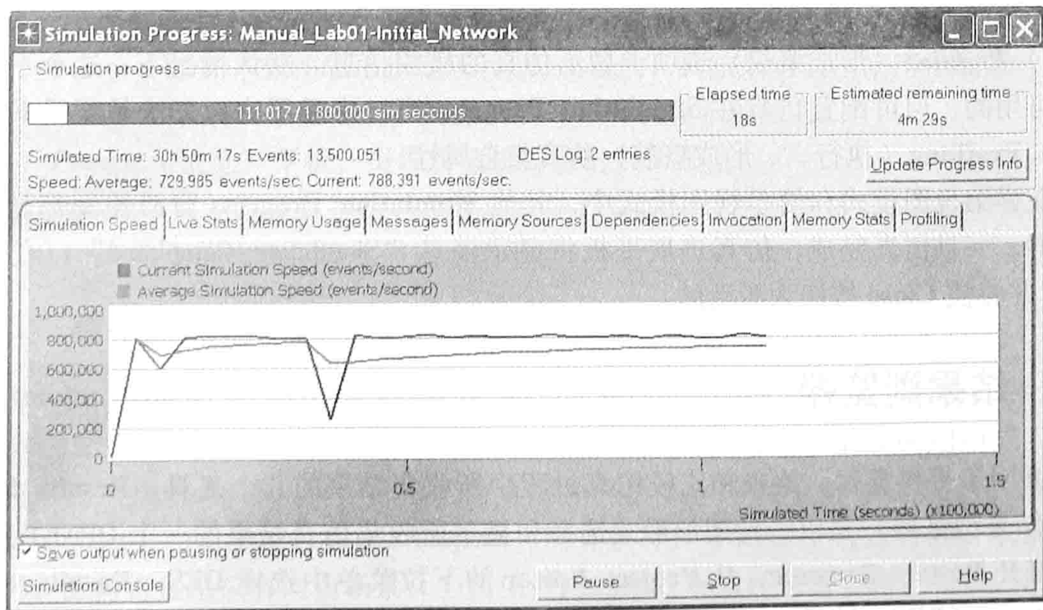


图4.9　**Simulation Execution** 窗口的详细视图模式

1） *Simulation speed*（仿真速度）包含一个图形，显示以事件数/s 为单位表示的平均仿真速度。

2） *Live Stats*（实况统计）选项卡可显示在仿真过程中正在被收集的某些统计量。为了在 *Live Stats* 平板中显示一个统计量，需要将仿真配置为收集这个统计量，以及将这个统计量配置为一个实况统计量，方法是在 **Choose Results**（选择结果）窗口的 *Data Collection*（数据收集）平板中的 *Generate live statistic*（产生实况统计量）检查框上单击（见 4.2.5 节）。注意，仅有全局和个体对象统计量（即通过在单个对象上右击打开 **Choose Results** 窗口）可被配置为作为实况统计量进行产生（即 OPNET 不能为多个节点产生实况统计量）。

3） *Memory Usage*（内存使用情况）选项卡显示当前仿真的内存消耗。

4） *Messages*（消息）选项卡显示由仿真产生的消息，描述自动化仿真进展的那些消息除外。

5） *Memory Sources*（内存源）和 *Memory Stats*（内存统计）显示有关 OPNET 所管理的内存块的详细信息。默认情况下这些平板是不可用的。需要配置仿真集合属性 **Runtime Displays... Memory Usage**（运行时显示 ... 内存使用情况），使这些平板在 **Simulation Progress** 窗口中成为可用的。

6） *Dependencies*（依赖关系）选项卡显示由仿真打开的所有文件的列表。默认情况下这个平板也是不可用的，但可配置仿真在 **Simulation Progress** 窗口中显示它，方法是通过在 **Runtime Display... Other Displays**（运行时显示 ... 其他显示）仿真集合属性下选择 *Generate list of component file dependencies*（产生组成文件依赖关系列表）检查框。

7） *Invocation*（命令触发启动）选项卡显示执行当前仿真的命令提示触发启动和所有相关联的参数。

8） *Profiling*（形成概貌）选项卡显示仿真的概貌信息。默认情况下，这个平板也是不可用的，但可配置仿真在 **Simulation Progress** 窗口中显示它，方法是配置 **Execution... Profiling**（执行 ... 形成概貌）仿真集合属性。

在详细视图模式和简单视图模式中，其他 **Simulation Progress** 窗口控制都是完全相同的。一旦仿真完成，仿真进展平板将显示消息 “Simulation Completed” （仿真完成），并将使 **Close** 按钮为激活的。

4.4 结果浏览器

OPNET 提供显示、检查和比较仿真过程中所收集结果的几个工具。**Results Browser**（结果浏览器）是以一种图形形式选择和显示所收集仿真结果的一个 OPNET 工具。为了打开 **Results Browser**，从 **Project Editor** 的下拉菜单中选择 **DES→Results→View Results...** （DES→结果→查看结果 ...）选项或简单地单击 *View Results*（查看结果）图标（）。

如图 4.10 所示，**Results Browser** 窗口包含三个选项卡：*DES Graphs*（DES 图形）、*DES Parametric Studies*（DES 参数化研究）和 *DES Run Tables*（DES 运行列表）（选项卡的标题页包含在括号中的一个数字，指明仿真运行次数）。*DES Graphs* 选项卡提供创建所收集统计量各种视图并以标准图形的形式显示它们的工具。*DES Parametric Studies* 选项卡支持统计量间的比较。另外，如果仿真有属性配置了多个值，则结果是，它执行了多次，那么 *DES Parametric Studies* 选项卡包含详细研究一个特定统计量如何随属性值的变化而改变的选项。最后，*DES Run Tables* 选项卡包含输出表报告，并依据这些输出表提供产生 Web 报告的一个选项。

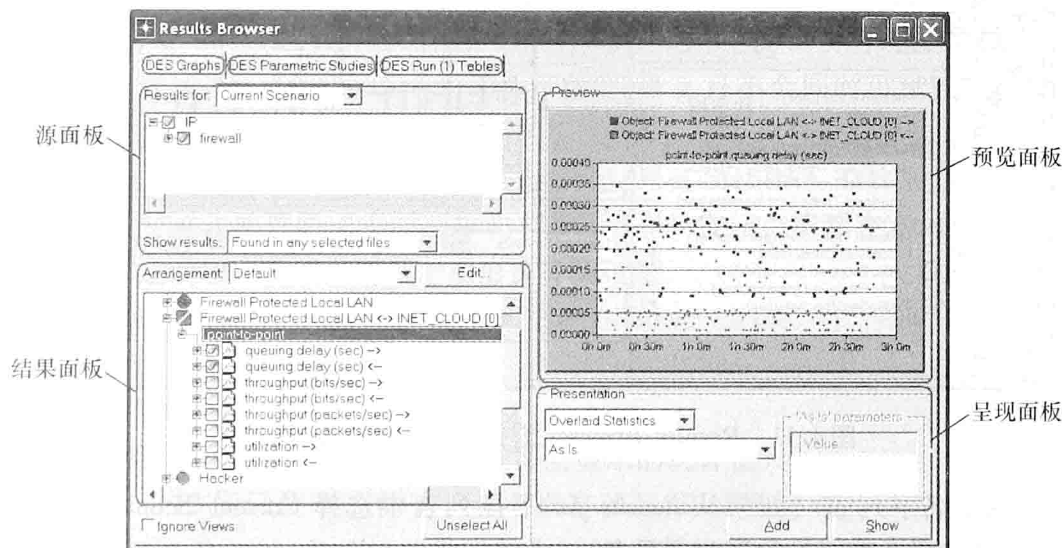


图 4.10　**Results Browser** 窗口

本节主要将讨论焦点放在通过 *DES Graphs* 选项卡可用的结果报告特征上面。*DES Graphs* 选项卡由四个不同的平板组成，如图 4.10 所示：

1）*Source*（源）平板使您可指定项目和/或场景，其结果将被用来产生图形。

2）*Results*（结果）平板是选择的那个统计量将被显示的地方。

3）*Preview*（预览）平板，提供所选统计量的一个预览。

4）*Presentation*（呈现）平板，配置所选统计量如何在图形中显示。

4.4.1 ~ 4.4.3 节比较详细地描述这些平板。

4.4.1　结果浏览器中的源平板

Results Browser 的 *DES Graphs* 选项卡中的 *Source* 平板由两个下拉列表［称为 *Results for*（结果）和 *Show Results*（显示结果）］和可用项目与场景的源树图组成。通过位于窗口左上角的 *Results for*，可指定仿真结果将从 Current Scenarios（当前各场景）、Current Project（当前项目）或 All Projects（所有项目）中的哪个检索得到。取决于您的选择，源树将显示只有当前场景、当前项目内的所有场景还是所有存在项目内的所有场景的仿真结果。默认情况下，在源树中仅选择当前场景。

存在项目和场景的源树被组织成三个等级，如图 4.11 所示：

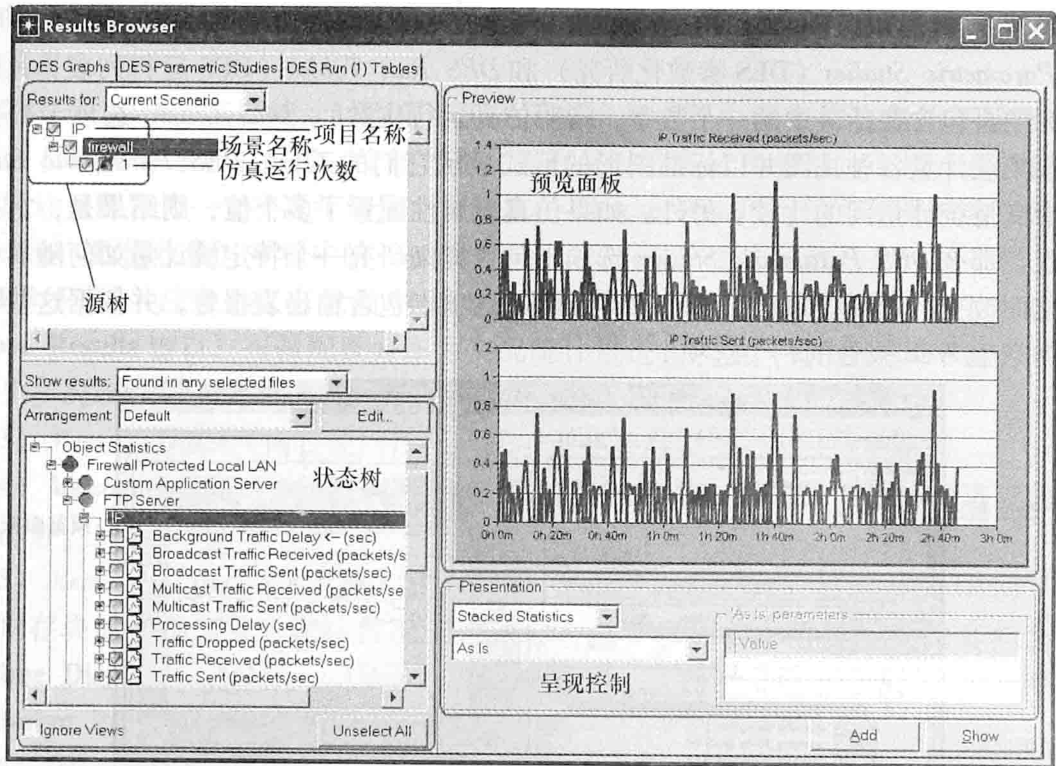


图 4.11　**Results Browser** 窗口的 DES Graphs 选项卡

1）项目名字列表（如果从 *Results for* 下拉列表中选择 Current Scenario 或 Current Project 选项，那么将仅显示当前项目名）。

2）每个项目的各场景名字列表。

3）每个场景的仿真运行列表。

通过在带有加号或减号的图标上单击，源树分支可进行展开或收缩。位于项目、场景或仿真运行名旁边的检查框，使您可选择或取消选择仿真结果的源。在项目的检查框上单击，可交替地选择和取消选择那个项目内的所有场景。类似地，在场景的检查框上单击可选择/取消选择那个场景内的所有仿真运行。

Show Results 下拉菜单位于 *Source* 平板底部。当选择了多个统计源时，它被用来确定在 *Statistics Tree*（统计树）将使用哪个统计源。这个列表有两个选项：

1）仅显示所有被选仿真运行都有的那些统计量。

2）显示在任意被选统计运行中存在的统计量。

因为每个仿真运行的结果都被保存到一个独立的文件，所以第一个选项称作 Common to all selected files（对所有被选文件都有的），第二个选项称作 *Found in any selected file*（在任意被选文件中找到的）。

4.4.2　结果浏览器中的结果平板

Results Browser 的 *Results* 平板使您可选择要显示的统计量。这个平板包含如下各项：

1）*Arrangement*（布局）：一个下拉菜单，指定在 **Results Tree** 内统计量如何组织。

2）*Arrangement* 列表旁边的 **Edit...** 按钮使您可进行统计布局的定制配置。

3）**Statistics Tree**（统计树），列出被选仿真运行的所有统计量。

4）*Ignore Views*（忽略视图）检查框指定针对隐藏的对象收集的统计量是否在 **Statistics Tree** 中显示。通过 **Project Editor** 中的 **View**（查看）下拉菜单，OPNET 提供一个选项，允许诸如链路、需求（demands）和域等对象成为不可见的。选择 *Ignore Views* 检查框，将包括所有对象的结果（包括在网络中隐藏的对象）。使这个检查框处于未被选择状态，将仅列出在网络中可见的那些对象的统计量。

5）**Unselect All**（取消选择所有的）按钮将使您以前选择的所有统计量都回到未必选择的状态。当选择了位于 **Statistics Tree** 的不同分支且不会同时可见的几个统计量时，这个选项是非常有用的。例如，显示第二个图形，就要求首先取消选择在第一个图形中已经显示的统计量，之后选择新的统计量。与浏览 **Statistics Tree** 搜索已经在第一个图形中使用的统计量的做法不同，单击 **Unselect All** 按钮，清除在 **Statistics Tree** 中的所有选择，这使您可直接为第二个图形选择统计量。

在 **Statistics Tree** 中浏览并选择统计量，与在 **Source Tree** 中执行的方法相同：

1）单击带有加号的图标，展开树的一个分支。

2）单击带有减号的图标，收缩该分支。

3）单击检查框选择/取消选择条目（即要显示的统计量）。

考虑如图 4.12 所示的一个例子，其中来自项目 IP 的场景 firewall 使 *Remote HTTP client* 节点的 **TCP Receive Buffer** 属性配置为 8760 字节和 32 768 字节两个值。这样的一个配置将仿真执行两次（即 **TCP Receive Buffer** 属性的每个值执行一次），并将仿真结果存储在两个独立文件中。OPNET 在这样的仿真运行之间做出区分的方法是，为每次运行创建一个唯一的名字，组成为 <*Project Name*> - <*Scenarios Name*> - *DES* - <*run number*>（<项目名> - <场景名> - DES - <运行号>）。因此，在上面的情况中，仿真运行将分别被命名为 *IP - firewall - DES - 1* 和 *IP - firewall - DES - 2*。

如果选择要显示的一个统计量，如 **FTP Server. Server Performance. Load（requests/sec）**，那么 OPNET 将自动地为那个场景内的所有仿真运行选择结果。

4.4.3 结果浏览器中的预览和呈现平板

Preview 平板是 **Results Browser** 窗口的第三个平板。这个平板自动地显示在 **Statistics Tree** 中选择的统计量。在这个平板中统计量如何显示，取决于在 *Presentation*（呈现）平板中做出的选择。

Presentation 平板是 **Results Browser** 窗口中的第四个也是最后一个平板。这个平板包含两个下拉列表。第一个下拉列表给出两个选项，当选择多个统计量时，它们控制图形的显示：

1）Stacked Statistics（堆叠的统计量）将每个被选择的统计量放到一个独立的图形中。对于多个统计量，这些图形的堆叠方式是一个堆叠在另一个之上，如图 4.11 和图

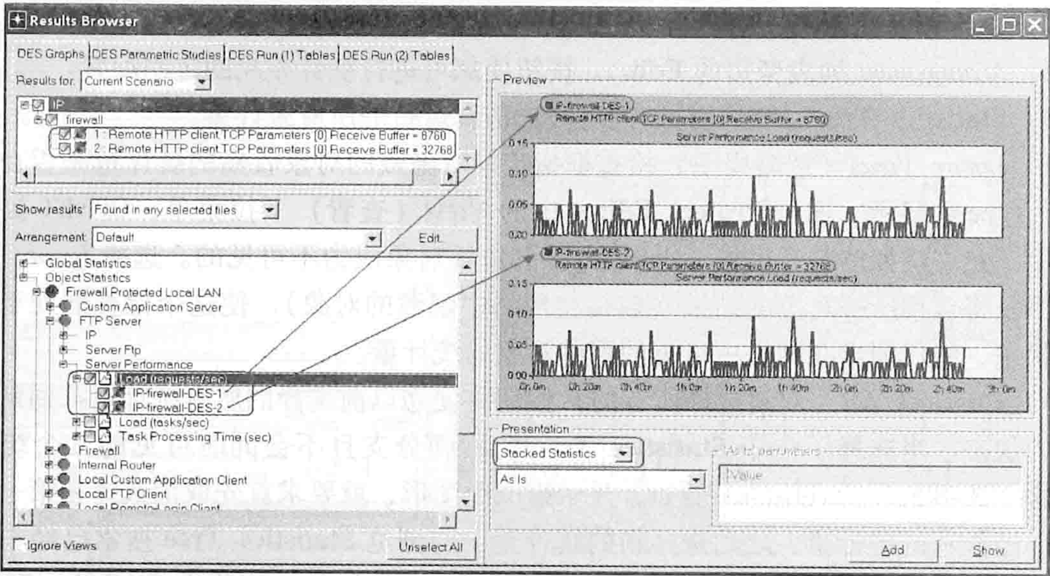


图 4.12　同一场景多次运行的仿真结果

4.12 所示。

2）Overlaid Statistics（覆叠的统计量）将每个被选择的统计量都放到同一图形中，参见图 4.10。当选中这个选项时，图形中的每个统计量以一种不同颜色画出。

Presentation 平板中的第二个下拉列表给出一个预设过滤器集合，可被选择来显示仿真结果。图 4.13 提供可用过滤器的一个完全列表，而下面的列表则提供最常用过滤器的一个简短描述。考虑如下表示法：V_i^k 是第 k 个被选中统计量的第 i 个值（值是从 0 开始计数的），T_i^k 是 V_i^k 被收集时的时间，且 S 是被选中统计量的总数，那么：

1）*As Is*（原样显示）过滤器，就像在仿真过程中已经收集到统计结果一样来显示它们。

2）*adder*（加法器）过滤器显示一个图形，它包含一个或多个统计量各值的和。由 *adder* 过滤器显示的第 i 个值如下计算：

$$adder(i) = \sum_{k=1}^{S} V_i^k$$

3）*average*（平均）过滤器为一个被选中的统计想显示各值的移动平均（running mean）。由第 k 个被选中统计量的 average 过滤器显示的第 i 个值是如下计算的：

$$average(i) = \frac{\sum_{j=0}^{i} V_j^k}{(i+1)}$$

As Is
Probability Density (PDF)
Cumulative Distribution (CDF)
Probability Mass (PMF)
Histogram (Sample Distribution)
Histogram (Time Distribution)
abscissa_filter
adder
average
constant_shift
delay_element
differentiator
exponentiator
gain
glitch_notch
integrator
limiter
logarithm
moving_average
multiplier
reciprocal
sample_sum
time_average
time_window
value_notch

图 4.13　预设过滤器的列表

4）*time_ average*（时间平均）为一个被选中的统计量显示统计值的移动连续平均（running continuous average）。实际上，每个值是由统计量具有那个值时的时间量加权的。第 k 个被选中统计量的 *time_ average* 过滤器显示的第 i 个值是如下计算的：

$$time_\ average(i) = \frac{\sum_{j=0}^{i-1}(V_j^k \times (T_{j+1}^k - T_j^k))}{\sum_{j=0}^{i}(T_{j+1}^k - T_j^k)}$$

5）*sample_ sum*（样本和）过滤器显示统计值的移动总和（running total）。第 k 个被选中统计量的 *sample_ sum* 过滤器显示的第 i 个值是如下计算的：

$$sample_\ sum(i) = \sum_{j=0}^{i} V_j^k$$

参考 OPNET 文档的预设过滤器章节，了解显示统计结果的所有预设过滤器模型的更多信息。

4.4.4　分析平板

一旦在 **Results Browser** 中选择必要的统计量及其呈现风格，那么在 **Results Browser** 窗口右下的 **Show** 按钮上单击，就创建了称为 **analysis panel**（分析平板）的一个新窗口，它包含被选中统计量的图形。图 4.14 给出依据如图 4.12 所示的选择所生成仿真图的一个分析平板。

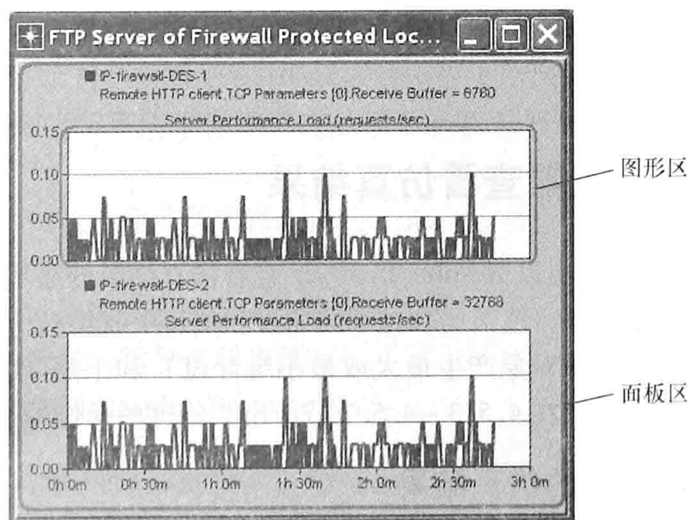
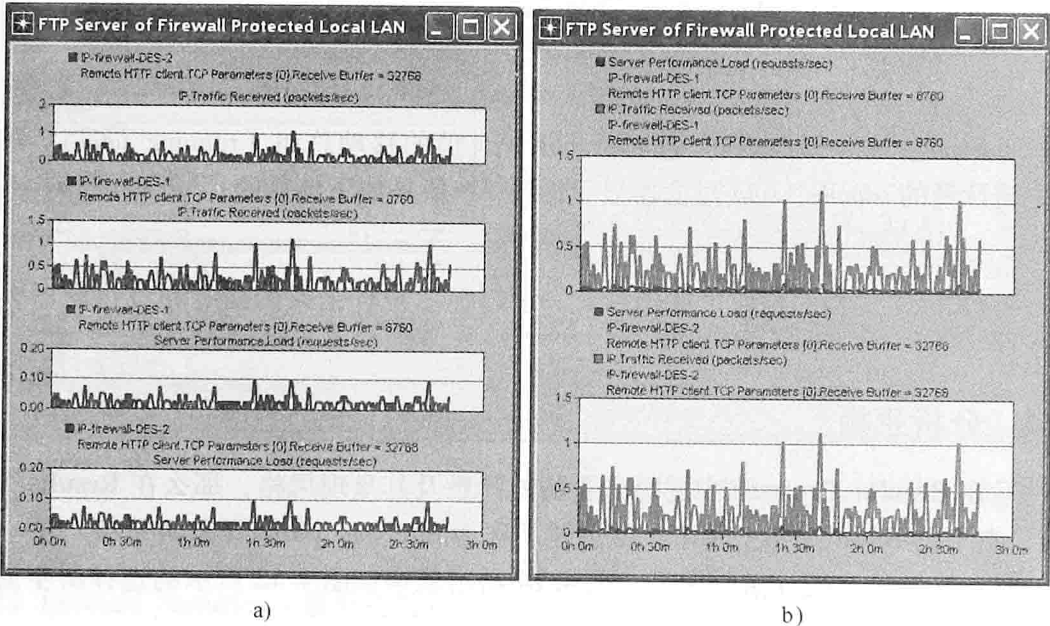


图 4.14　在图 4.12 中选中统计量的分析平板

位于 **Results Browser** 窗口底部 **Show** 按钮左侧的 **Add** 按钮，支持将选中的统计量添加到一个已经创建的分析平板中。需要做的所有事情就是选择关注的统计量，单击 **Add** 按钮，之后在分析平板中单击。但是，取决于在分析平板中单击的位置，被选中的统计量添加的方式就不同了。分析平板由两部分组成：图形区（以白色显示）（其中实际的仿真结果被画为一个图形）和平板区（以灰色显示）（在所有四侧形成图形区的边界）。如果在单击 **Add** 按钮之后，在平板区（灰色）单击，那么被选中统计量的添加方式，就像已经选择 *Stacked statistics* 呈现选项一样，即在相同平板内为这些统计量画出一

个独立的图形。另外，如果在图形区（白色）中单击，那么被选中统计量被添加到现有图形的方式就像已经选择 *Overlaid Statistics* 呈现选项一样。图 4.15 给出在使用这两种方法将新的统计量添加到如图 4.14 所示的分析平板中之后分析平板显示的样子。



a)　　　　b)

图 4.15　将新的统计量添加到图 4.14 已有的分析平板

a）在平板区单击，使用 *Stacked Statistics* 呈现选项　b）在图形区单击，使用 *Overlaid Statistics* 呈现选项

4.5　使用结果浏览器查看仿真结果

4.5.1 节和 4.5.2 节给出以 **Results Browser** 查看仿真结果的逐步骤的指令。OPNET 也提供了几个其他功能特征，包括比较在不同场景和项目中收集的仿真结果、寻找极值结果（即在仿真中寻找哪个对象产生最大或最小统计值）和采用 **Time Controller**（时间控制器）查看仿真结果。在 4.5.3 ~ 4.5.7 节给出实施这些操作的指令。

4.5.1　查看当前场景的仿真结果

1）打开 **Results Browser** 窗口：

方法 1：在 **Project Editor** 的项目工作空间的任何地方右击，并从下拉菜单中选择 **View Results**（查看结果）。

方法 2：从 **Project Editor** 的下拉菜单中选择 **DES→Results→View Results...**（DES→结果→查看结果...）。

方法 3：在 **Project Editor** 中，单击 *View Results*（查看结果）图标（）。

2）在 **Results Browser** 窗口中，选择要显示的统计量及其呈现。

3）单击 **Show** 按钮（这将不会关闭 **Results Browser** 窗口），显示带有所选择统计量图形的分析平板。

4）单击 **Unselect All**（取消选择所有的）按钮清除统计量选择（可选的）。

5）依据必要的次数，重复这些步骤。

6）当完成时，关闭 **Results Browser** 窗口。

4.5.2 查看网络中一个特定对象的仿真结果

1）选择关注的对象，之后右击它。

2）从弹出菜单中选择 **View Results**。

3）出现的 **Results Browser** 窗口，将仅包括为所关注对象收集的统计量。

4）选择要显示的统计量及其呈现。

5）单击 **Show** 按钮，显示带有所选择统计量图形的一个平板。

6）单击 **Unselect All** 按钮清除统计量选择（可选的）。

7）依据必要的次数，重复这些步骤。

8）当完成时，关闭 **Results Browser** 窗口。

4.5.3 查看这个项目和其他项目中各场景的仿真结果

1）打开 **Results Browser** 窗口（见 4.5.1 节）。

2）从称为 *Results for* 的下拉菜单中，选择 Current Project（当前项目）（如果希望详细研究当前项目各场景中收集的统计信息）或选择 All Projects（所有项目）（如果希望详细研究所有存在项目的所有场景中收集的统计信息）。

3）**Source Tree** 显示可用于详细研究的项目和场景的列表。选择希望详细研究结果的场景。

4）选择要显示的统计量及其呈现。

5）单击 **Show** 按钮，显示带有所选择统计量图形的一个平板。

6）单击 **Unselect All** 按钮清除统计量选择（可选的）。

7）依据必要的次数，重复这些步骤。

8）当完成时，关闭 **Results Browser** 窗口。

4.5.4 比较仿真结果

1）打开 **Results Browser** 窗口，*Results for* 下拉菜单设置为 Current Project 或 All Projects：

方法 1：打开 **Results Browser** 窗口（见 4.5.1 节），并将 *Results for* 下拉菜单设置为 Current Project 或 All Projects。

方法 2：从 **Project Editor** 的下拉菜单中选择 **DES→Results→Compare Results...**（DES→结果→比较结果...）。

2）**Source Tree** 显示可用于详细研究的项目和场景的列表。选择希望比较结果的场景。

3）选择要显示的统计量及其呈现。

4）单击 **Show** 按钮，显示带有所选择统计量图形的一个平板。

5）单击 **Unselect All** 按钮清除统计量选择（可选的）。

6）依据必要的次数，重复这些步骤。

7）当完成时，关闭 **Results Browser** 窗口。

4.5.5 将新的统计量添加到现有图形

1）打开 **Results Browser** 窗口，并显示带有所选择统计量的一个平板（见 4.5.1 节）。

2）选择要显示的其他统计量及其呈现。

3）单击 **Add** 按钮，将所选中的统计量添加到一个现有分析平板。

4）单击希望添加所选统计量的图形（在图形区或平板区）。

5）依据必要的次数，重复这些步骤。

6）当完成时，关闭 **Results Browser** 窗口。

4.5.6 寻找最大结果

1）打开 **Select Statistics for Top Results**（为最大结果选择统计量）窗口（见图 4.16a）：

方法 1：在 **Project Editor** 的项目工作空间内的任何地方右击，并从弹出菜单中选择 **Find Top Results**（寻找最大结果）。

方法 2：从 **Project Editor** 的下拉菜单中选择 **DES→Results→Find Top Statistics...**（DES→结果→寻找最大统计量...）。

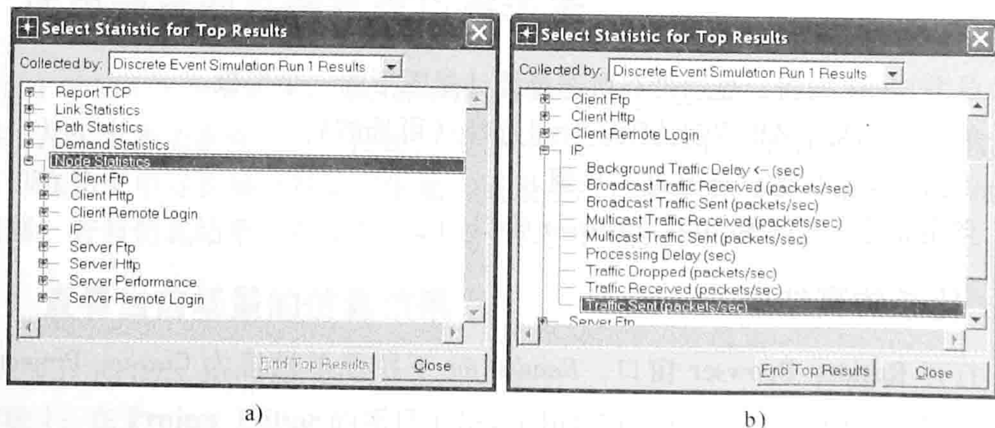


图 4.16 **Select Statistics for Top Results** 窗口

a）展开类 *Node Statistics* b）选择 *IP* 类中的 **Traffic Sent**（**packets/sec**）

2）在 **Select Statistics for Top Results** 窗口中，选择希望为之寻找最大结果的统计量（见图 4.16b）。

3）可能也希望指定希望详细研究哪个仿真运行，方法是从图 4.16a 所示的 *Collected by* 下拉列表中选择所提供的选项之一。

4）当选择了关注的统计量时，**Find Top Results** 按钮将成为可用的。单击 **Find Top Results** 按钮，打开 **Top Objects** 窗口。

如图 4.17 所示的 **Top Objects** 窗口，为网络中所有可适用的对象列出了选择的统计值（即被选中的统计量被配置为收集的对象）。可进一步配置最大结果图，方法是指定显示条件和满足那个条件的最大统计数量，如 *MaxN*。*MaxN* 是一个正数，确定要被显示的最大统计数量。*Display condition*（显示条件）指定一项准则，确定一个特定统计量是否能够被显示。准则是如下指定的，通过列出要被比较的统计值的一个函数（即最小、平均、最大或标准差）、比较符号（即大于“>”或小于“<”）和统计值将与之比较的实数。仅有满足显示条件的那些统计量才被显示在最大结果图形中，这意味着将显示小于 *MaxN* 个统计量是可能的。

图 4.17 所示是 **Top Objects** 窗口的一个例子，它列出包含被选中的统计量 **IP. Traffic Sent（packets/sec）**的 16 个对象。但是，Top Objects 窗口被配置为至多显示平均值大于 0 的 10 个统计量。即使所有的 16 个统计量都具有大于 0 的平均值，也仅有 10 个统计量（其顺序依据平均值从最大到最小排序）将被显示在最大结果图形中，如预览平板所示。

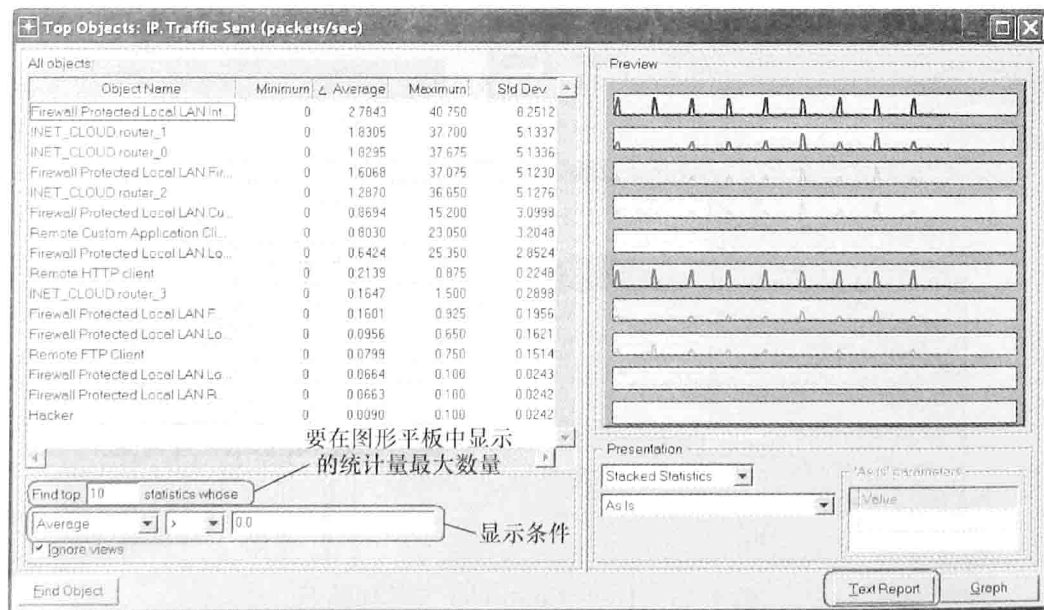


图 4.17　在图 4.16b 中所选中统计量的 **Top Objects** 窗口

在 **Top Objects** 窗口右下部的 **Graph** 按钮上单击，将创建一个最大结果分析平板，该平板有满足显示条件的至多 *MaxN* 个统计量。如果没有统计量满足显示条件，那么将不创建分析平板。最后，通过单击 **Text Report**（文本报告）按钮，可将有关最大统计的信息输出到一个文本编辑器。

4.5.7　采用时间控制器查看结果

OPNET 也提供称为 **Time Controller**（时间控制器）的一个选项查看仿真随时间的

进展情况。这个选项使您可查看在一段时间上网络对象（例如，如果对象正在移动的话）的变化和相应的统计结果。

1）在一个分析平板中显示所关注的统计量（见 4.4.4 节）。

2）打开 **Time Controller** 工具：

方法 1：从 **Project Editor** 的下拉菜单中选择 **View→Show Time Controller**（查看→显示时间控制器）。

方法 2：按 Ctrl + Alt + T 键。

3）**Time Controller** 窗口将出现，如图 4.18a 所示。可使用回放控制（即到开始、回退、暂停、播放、前进和到结束）和加速/慢速按钮控制仿真回放。为进一步配置回放，也可单击 **Configure** 按钮，打开 **Time Controller Settings**（时间控制器设置）（见图 4.18b）。

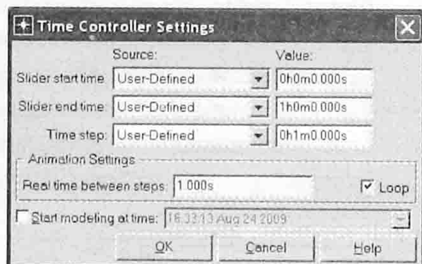
4）单击 **Play** 按钮（▷）开始仿真回放。一旦回放开始，一条绿线（显示仿真进度）将出现在被显示的图形中（见图 4.18c）。

5）单击 **Help** 按钮，了解有关 **Time Controller** 窗口中控制的更多信息。

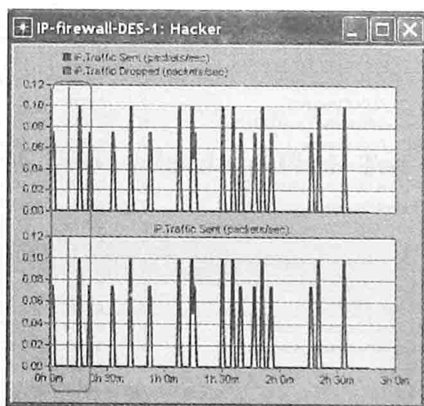
6）一旦完成，单击 **Close** 按钮退出 **Time Controller** 工具。



a)



b)



c)

图 4.18 **Time Controller** 工具的例子

a）在 **Time Controller** 窗口中的控制 b）配置选项的 **Time Controller Settings** 窗口

c）一条绿线在被显示的图形间移动，显示仿真随时间的进度

4.6 管理分析平板

OPNET 提供了一个操作集合，来操控一个被显示的分析平板，包括显示/隐藏打开的分析平板、以最新的仿真结果更新分析平板、从图形中去除所显示的统计量、修改分析平板外观和详细研究显示在图形中的原始统计数据。本节提供有关实施操控被显示图形的这些和其他操作的指令。

4.6.1　隐藏/显示分析平板

一旦显示了分析平板，就可使用这些操作来隐藏它们或再次显示它们。可在所有平板上或在个体平板上做到这一点。

下面是隐藏/显示所有分析平板的步骤：

方法 1：单击 **Project Editor** 中的 *Hide/Show Graph Panels*（隐藏/显示图形平板）图标（）。单击这个图标，将隐藏或显示所有的分析平板。

方法 2：从 **Project Editor** 的下拉菜单中选择 **DES→Panel Operations→Arrange Panels→Show All**（DES→平板操作→安排平板→显示所有的）将显示所有的分析平板，或 **DES→Panel Operations→Arrange Panels→Hide All**（DES→平板操作→安排平板→隐藏所有的）将隐藏所有的分析平板。

使用如下步骤，隐藏/显示个体分析平板：

方法 1：从 Project Editor 的下拉菜单中选择 **DES→Panels→<analysis panel name>**（DES→平板→<分析平面板名>），其中 **analysis panel name** 是希望隐藏/显示的平板的名字。Panels（平板）下拉菜单列出当前被显示的那些平板的所有可用分析平板，在其名字旁边包含一个检查点。选择这个菜单选项将交替地显示或隐藏被选的平板。图 4.19 形象地给出一种情况，其中显示的是名为 *Packets Dropped by the Firewall*（由防火墙丢弃的报文）的分析平板。

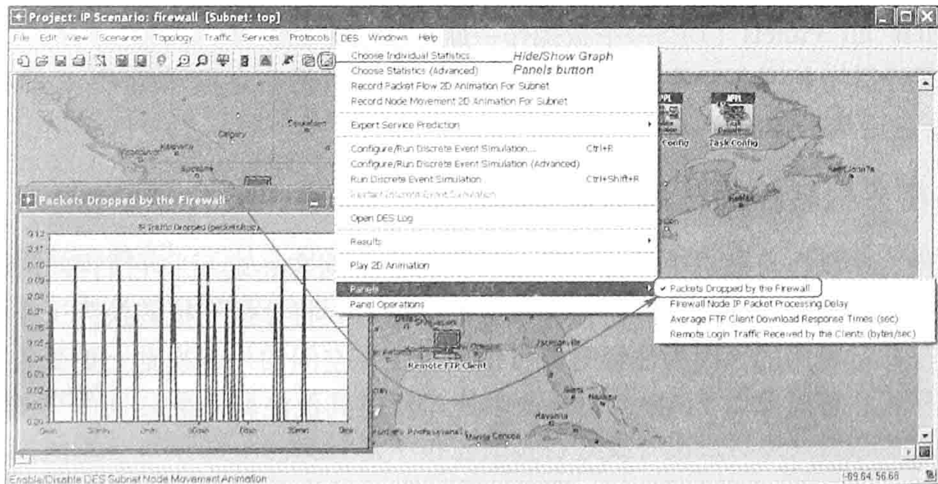


图 4.19　隐藏/显示分析平板

方法 2：在分析平板的平板空间（灰色区）内右击，并从弹出菜单中选择 **Hide This Panel**（隐藏这个平板）。

方法 3：在分析平板窗口的右上角单击 *Close* 图标（即交叉符号）。之后在弹出的平板上单击 **Hide** 按钮。

4.6.2　安排分析平板

Project Editor 的 **DES→Panel Operations**（DES→平板操作）菜单包含其他平板操

作的一个列表，如图4.20所示。**Panel Operations→Arrange Panels**（平板操作→安排平板）选项使得可在屏幕上安排当前可见的平板：**Distribute**（分布）和**Tile**（瓦盖形）选择确保平板显示时不会覆盖其他平板，且均匀地分布在**Project Editor**的工作空间的区域上，而**Cascade**（级联形）选项以一种级联方式在另一个的上面显示平板。

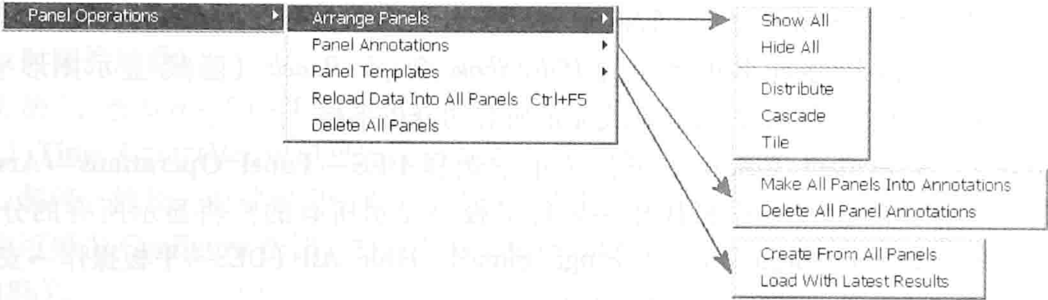


图4.20 **Panel Operations**（平板操作）下拉菜单中的可用选项

4.6.3 删除分析平板

可单个地或一次全部地删除分析平板。这个操作物理上去除一个被选中的平板或创建的所有分析平板。被删除的分析平板不能再被查看，如果希望再次查看，则必须重新创建。

为删除所有的分析平板：从**Project Editor**的下拉菜单中选择**DES→Panel Operations→Delete all Panels**（DES→平板操作→删除所有平板）。

为删除个体分析平板：在希望删除的分析平板窗口的右上角单击*Close*图标（即交叉符号）。之后单击弹出窗口中的**Delete**按钮。

4.6.4 将平板转换为注释对象

可使用这些操作将被选中的分析平板转换为注释对象，之后将它们再转换回平板。当准备一个讲稿来描述一项仿真研究和相应的结果时，将分析平板转换为工作空间内的注释对象是非常有用的，因为注释对象可使项目工作空间的外观视觉上更受人喜欢并容易描述。平板注释对象以**Project Editor**中处理任何其他注释对象相同的方式（见2.9节）被处理。

如图4.20所示，**Panel Annotations**（平板注释）下拉菜单选项提供两个选择：

1）**Make All Panels Into Annotations**（将所有平板转换到注释）将所有分析平板内嵌在当前场景中，包括当前没有显示的那些平板，将它们转换为注释对象。

2）**Delete All Panel Annotations**（删除所有平板注释）将所有平板注释对象转换回常规分析平板。

也可将被选中的分析平板转换为注释对象，方法是在分析平板的平板空间（灰色区）内右击，从弹出的菜单中选择**Make Panel Annotation In Network**（制作网络中的平板注释）。在平板注释对象上双击，将之转换回常规平板图形。平板注释对象处理的方式与Project Editor中处理任何其他注释对象的方式相同。

下面是将所有分析平板转换为注释对象的步骤：从 **Project Editor** 的下拉菜单中选择 **DES→Panel Operations→Panel Annotations→Make All Panels Into Annotations**（DES→平板操作→平板注释→将所有平板制作为注释）。

为将个体平板制作为注释对象，使用如下方法之一：

方法 1：在分析平板的平板空间（灰色区）内右击，并从弹出菜单中选择 **Make Panel Annotation In Network**（制作网络中的平板注释）。

方法 2：在分析平板空间内双击。

删除所有注释对象的步骤如下：从 **Project Editor** 的下拉菜单中选择 **DES→Panel Operations→Panel Annotations→Delete All Panels Annotations**（DES→平板操作→平板注释→删除所有平板注释）。

为删除个体注释对象，使用如下方法之一：

方法 1：在平板注释对象上双击。一旦移动新恢复的分析平板，会仍然观察到 **Project Editor** 的工作空间内带有标签 *Panel Open* 的注释对象。选择那个注释对象，并在键盘上按 Delete 键。

方法 2：从 **Project Editor** 的下拉菜单中选择 **DES→Panel Operations→Panel Annotations→Delete All Panels Annotations**（DES→平板操作→平板注释→删除所有平板注释）。

4.6.5　以新的结果重新载入分析平板

另一项有用的操作是以一次新的仿真运行的结果，重新载入现有的分析平板。考虑这样一种情况，其中可能创建了带有各种统计图的几个平板，目的是分析一次仿真运行的结果。在关闭所收集结果的详细研究时，您意识到仿真研究的参数之一没有正确配置，且仿真需要重新运行。一旦收集了新的仿真结果，希望像以前一样详细研究相同的图，但要用新的数据。与从头开始重新创建所有的分析平板的做法不同，可将分析平板转换为模板（即没有任何数据的分析平板），并将新得到的结果重新载入这些平板模板中。

如果一些统计量已经显示在一个分析模板中，且后来改变了在仿真过程中要收集哪些统计量，从而一些要显示的统计量不再选择进行收集，那么以最新的仿真结果重新载入分析模板将失败。如果所收集的仿真结果被删除（如通过使用 **Manage Scenarios** 选项），则重新载入也将失败，且没有新的结果针对那个场景进行收集。

使用如下步骤，以最新的仿真结果重新载入所有的分析平板：

1）在不关闭现有分析平板的条件下，重新运行仿真。

2）使用如下三种方法之一以新结果重新载入分析平板：

方法 1：

① 从 **Project Editor** 的下拉菜单中选择 **DES→Panel Operations→Panel Templates→Create From All Panels**（DES→平板操作→平板模板→从所有平板创建）。这将所有分析平板（隐藏的或显示的）转换为图形模板。

② 从 **Project Editor** 的下拉菜单中选择 **DES→Panel Operations→Panel Templates→Load With Latest Results**（DES→平板操作→平板模板→以最新结果载入）。这会以最新的仿真数据更新所有的图形模板。

方法 2：从 **Project Editor** 的下拉菜单中选择 **DES→Panel Operations→Reload Data Into All Panels**（DES→平板操作→将数据重新载入所有平板）。这个选项首先从所有平板创建平板模板，之后以最新的仿真结果载入所有的模板平板。

方法 3：在键盘上按 Ctrl + F5 键。

使用如下步骤，以最新仿真结果重新载入个体模板：

1）重新运行仿真。

2）在分析平板的任何地方右击，并从弹出菜单中选择 **Make Panel Into Template**（将平板制作成模板），这将分析平板转换为一个模板平板。

3）在分析平板的任何地方右击，并从弹出菜单中选择 **Load Data Into Template**（将数据载入模板），这将最新的仿真数据载入平板。

4.7 高级分析平板性质

不仅以一个图形的形式查看统计结果，而且详细研究原始数值数据，通常是重要的。OPNET 支持将统计数据输出到一个电子表格（spreadsheet）或通过一个内部编辑器程序查看数据。改变输入如何在平板内被查看的这些操作和其他操作，可通过分析平板中的弹出菜单进行。

每个分析平板由两个区组成：图形区和平板区，如图 4.14 所示。图形区显示实际的图形，并具有一个白色背景。平板区是灰色的，并在所有四侧形成图形区的边界。在平板区和图形区右击相应地产生不同的弹出菜单，如图 4.21 所示。图 4.21a 给出弹出菜单，在右击平板区时打开；图 4.21b 给出的弹出菜单，当右击图形区时出现。在 4.7.1 节和 4.7.2 节描述这些菜单。

4.7.1 平板区弹出菜单

panel area pop-up menu（平板区弹出菜单）（见图 4.21a）主要处理在平板本身上的操作，包括如下方面：

1）**Edit Panel Properties**（编辑平板性质）——使您可改变平板性质，包括平板标题、背景色、平板在屏幕上的位置等。

2）**Show Statistic Data**（显示统计数据）——打开 **OPNET editor** 窗口，它包含原始统计数据及其在这个平板中显示的统计项的汇总（summary）。

3）**Duplicate This Panel**（复制这个平板）——创建同一分析平板的另一个备份。

4）**Add Graph**（添加图形）——打开 **Results Browser** 窗口，这使您可选择其他统计量，添加到分析平板（也可见 4.5.5 节）。

5）**Use Same Vertical Scale**（使用相同的垂直尺度）——在平板内调整所有图形

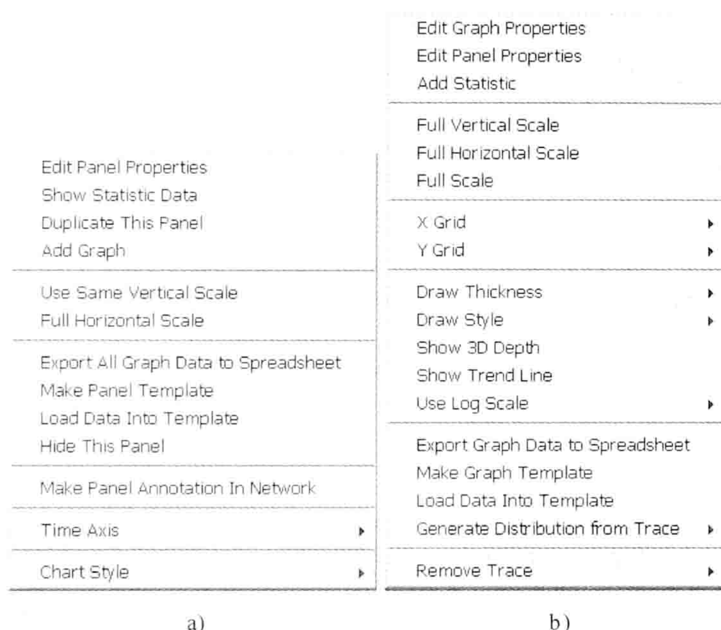


图 4.21 分析平板弹出菜单
a）当右击平板区时 b）当右击图形区时

（如果存在一个以上的图形），使它们具有相同的垂直尺度。

6）**Full Horizontal Scale/Full Vertical Scale/Full Scale**（全水平尺度/全垂直尺度/全尺度）——恢复图形的原水平/垂直/两尺度。

7）**Export All Graph Data to Spreadsheet**（将所有图形数据输出到电子表格）——将图形的原始统计数据输出到一个空格隔开的文本文件，该文件由用户配置的电子表格程序（如 Excel）自动打开。可改变用来打开所输出数据的电子表格程序，方法是改变 **spreadsheet_ prog** 首选项的值。

8）**Make Panel Template/Load Data Into Template**（制作平板模板/将数据载入模板）——使您以最新的仿真结果重新载入分析平板（见 4.6.5 节）。

9）**Hide This Panel**（隐藏这个平板）——从视图中隐藏该平板（见 4.6.1 节）。

10）**Make Panel Annotation In Network**（制作网络中的平板注释）——将平板转换为一个注释对象，并将之放入 **Project Editor** 的工作空间（见 4.6.4 节）。

11）**Time Axis**（时间轴）——使您可指定在分析平板中使用的仿真时间轴的类型。可能的时间轴显示选项包括 **Date/Time**（日期/时间）（如果存在的话）、**seconds**（秒）或 **auto - scale**（自动 - 尺度）（默认的）。

12）**Chart Style**（表风格）——使您可将分析平板的默认外观改变为预设配置之一，即 **Default**（默认的）、**Classic**（经典的）、**Gold Color Scheme**（金色方案）或 **Silver Color Scheme**（银色方案）。

Edit Panel Properties（编辑平板性质）选项打开一个 **Panel Operations**（平板操作）窗口，使您可改变一些平板性质。图 4.22 给出一个图，还有其相应的 **Panel Operations** 窗口，它包含如下选项：

1）*Panel title*（平板标题）文本框用于指定图形之平板的标题。

2）**Panel Coordinates**（平板坐标）按钮使您可设置平板的宽度、高度和在屏幕上的位置。

3）**Set Color**（设置颜色）按钮使您可指定平板的背景色，对于 *Default* 表风格，它默认是灰色的。

4）*Horizontal label*（水平标签）、*Horizontal min*（水平最小）、*Horizontal max*（水平最大）文本框和 **Full Scale**（全尺度）按钮使您可管理图形的水平轴的性质。

5）检查框 *Set all draw styles*（设置所有的画法风格）与下拉列表一起控制显示在图中的数据的画法风格。数据可使用 linear（线性）、sample - hold（样本保持）、bar（柱形）、discrete（离散）、square - wave（方波）或 bar chart（柱形图）风格，这在 4.2.6 节做了描述。

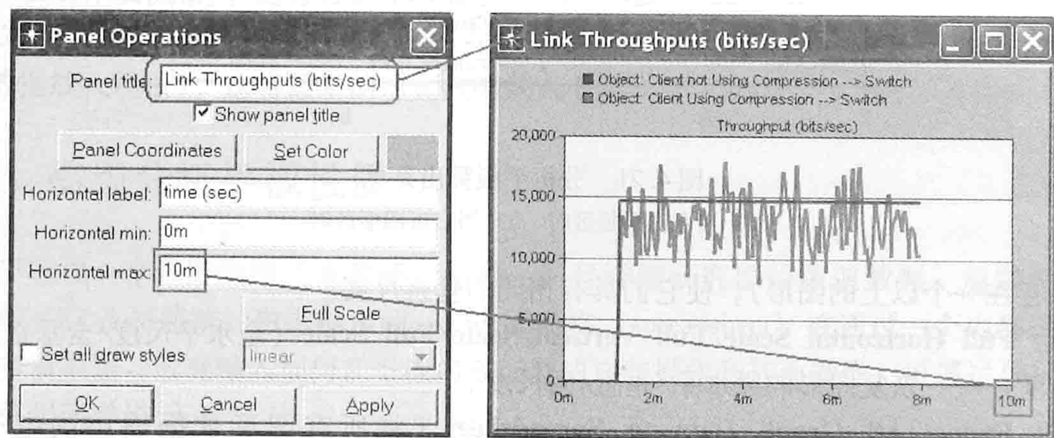


图 4.22 一个 **Panel Operations** 窗口及其图形

Show Statistic Data（显示统计数据）选项打开一个 **Statistic Information**（统计信息）窗口，包含原始图形数据及其汇总。通过一个下拉列表，**Statistics Information** 窗口（见图 4.23）提供两个选项：

1）General Statistic Info（一般统计信息）——所显示图形数据的一个汇总，包括 max（最大）、min（最小）、expected value（期望值）、sample mean（样本均值）、variance（方差）、confidence intervals（置信区间）和其他统计信息。

2）Statistic Data（统计数据）——被显示图形的原始统计值的一个列表。

Statistic Data 窗口是可编辑的。可在这个窗口中改变原始数据值，之后单击 **Build New Statistic**（显示新的统计）按钮，显示带有使用更新的统计结果值构造的一个新图形的平板。

4.7.2 图形区弹出菜单

如图 4.21b 所示的图形区弹出菜单，处理一个分析平板中数据图形的操作。在这个菜单中的一些选项对平板区弹出菜单是一样的，并已经在 4.7.1 节做了描述。下面是对在平板弹出菜单中不存在的那些选项的一个简单描述：

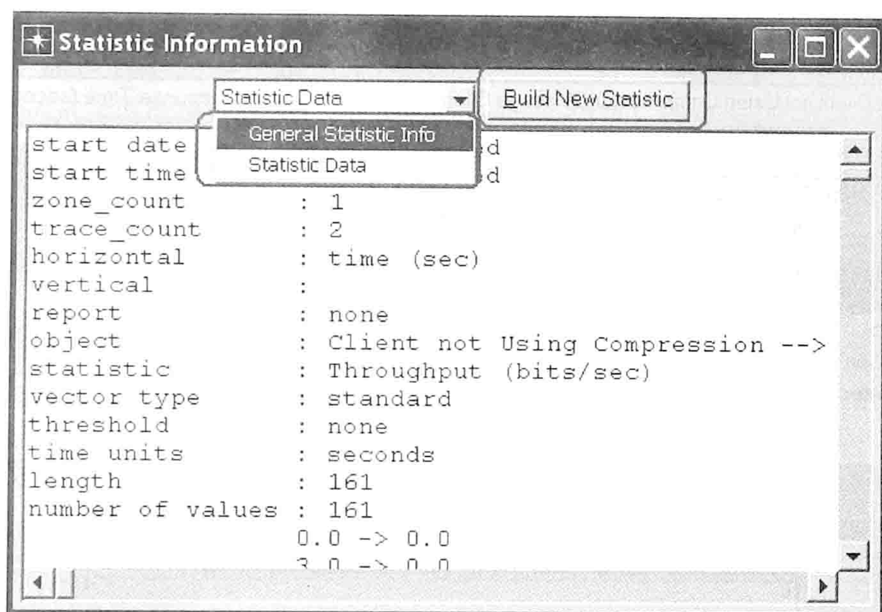


图 4.23　**Statistic Information** 窗口

1）**Edit Graph Properties**（编辑图形性质）——打开一个窗口，支持在图形中改变数据被显示的方式，包括诸如设置图例标题、改变画法风格、指定垂直最小和最大等操作。

2）**X Grid**，**Y Grid**（X 网格，Y 网格）——改变 X 和 Y 网格线如何被显示：Disabled（禁止）（没有线）、Solid（实线）或 Dashed（虚线）。默认情况下，X 轴网格线是不显示的（即禁止的），Y 轴网格线是使用一条实线显示的。

3）**Draw Thickness/Draw Style/Show 3D Depth/Show Trend Line/Use Log Scale**（画线宽度/画线风格/显示 3D 深度/显示趋势线/使用对数尺度）——使您可改变实际数据如何在图形中显示（例如，被显示数据线的宽度和画线风格，包括或去除 3D 深度和趋势线，以及使用对数尺度显示数据）。

4）**Generate Distribution from Trace**（从踪迹数据中产生分布）——依据图形的数据产生一个概率分布函数（PDF），并将结果保存到一个文件。默认情况下，文件被保存到 op_ models 目录。这个文件可使用 OPNET 的 PDF Editor（编辑器）打开，这使您可创建、更新和查看概率密度函数。

5）**Remove Trace**（去除踪迹数据）——使您可从图形中去除统计踪迹之一（即统计数据的一个集合）。

图 4.24 给出 **Edit Graph Properties**（编辑图形性质）窗口，这使您可改变图形的显示性质。这里不描述存在于这个窗口中的选项，因为它们非常类似于本节中描述的其他选项，且大部分是自解释的。

最后，偶尔在显示于图中的数据上放大是有益的。为了做到这一点，在图形中的任何地方单击，在按住鼠标键的同时，将鼠标在希望放大的区上拖过。选项 **Full Horizontal Scale**（全水平尺度）、**Full Vertical Scale**（全垂直尺度）或 **Full Scale**（全尺

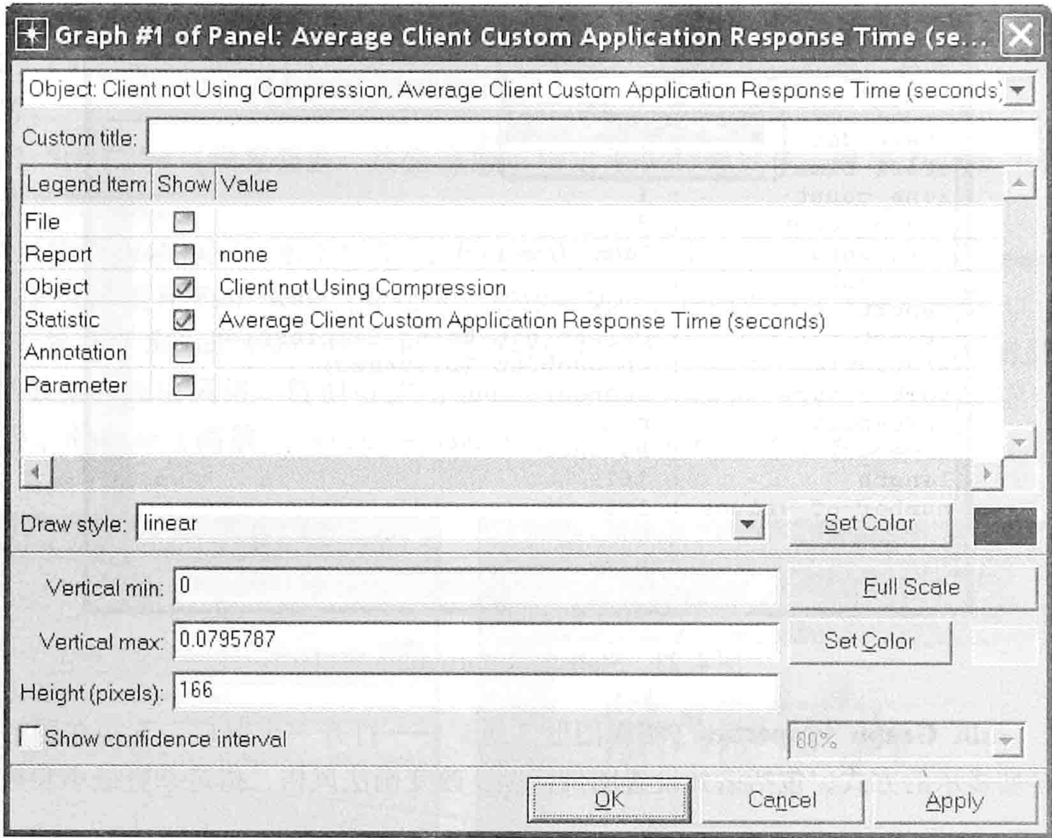


图 4.24 **Edit Graph Properties** 窗口

度）可恢复原图形视图。另外，可使用 **Edit Graph Properties**（编辑图形性质）和 **Edit Panel Properties**（编辑平板性质）改变垂直和水平最小和最大值，这将在被显示图形的特定部分取得放大或缩小的效果。

4.8 DES Log

当一个仿真执行时，它记录在仿真过程中发生的所有重要事件。这种事件的例子有仿真告警和错误、没有预料到的协议行为以及重要的或不正常的任何事件。这些事件被存储在称为 DES Log（日志）的一个纯 ASCII 文本文件中。DES 日志的文件名是 log_info，且它位于相应场景的结果文件夹的项目的目录中。通常情况下，结果文件夹以项目名开始，后跟场景名和 DES 运行号，都以一个短线隔开。例如，在 OPNET IT Guru Version 16.0 中，项目 HTTP 的一个场景名为 HTTP_ simple 的结果文件夹被命名为 HTTP - HTTP_ simple - DES - 1. olf. dir。

但是，不需要深入了解（peeking into）ASCII 文件，也许最好通过 OPNET 的 **Log Viewer** 工具详细研究 DES Log。通过从 **Project Editor** 的下拉菜单中选择 DES→Open Des Log（DES→打开 DES 日志），就可访问 **Log Viewer**（日志查看器）。如图 4.25 所示的 **Log Viewer** 窗口，由两个选项卡、几个平板和在窗口底部的一些控制组成。在

Log Viewer 中的选项卡数量取决于仿真配置和当前与主 OPNET 软件（即 Modeler、IT Guru 等）一起运行的模块，如 Flow Analysis（流分析）。在本节，将仅描述 *Discrete Event Simulation*（离散事件仿真）和 *Miscellaneous*（杂项）选项卡。

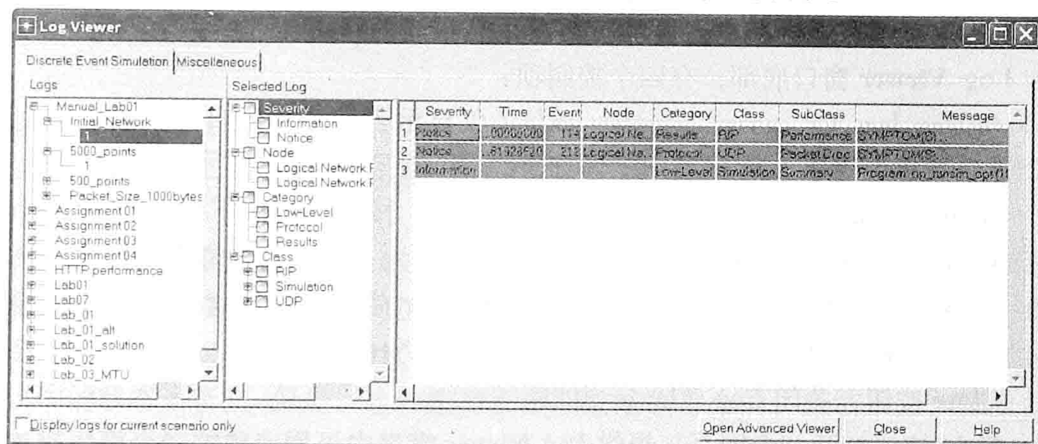


图 4.25　**Log Viewer** 窗口的 *Discrete Event Simulation*（离散事件仿真）选项卡

当调试一个仿真时，DES 日志是非常有帮助的，因为它提供了在仿真过程中发生了什么事件的深入了解，且它记录了当仿真正被执行时所发生的任何异常事件。每次仿真运行至少产生一些 DES 日志消息。当一次仿真运行产生三条或更少的 DES 日志消息，这典型地意味着仿真是配置正确的，且 OPNET 没有识别出任何事情出现异常。但是，如果有三条以上的 DES 日志消息，那么仔细地检查这些消息以确保仿真像预期一样的行为，就是一个不错的思路。

4.8.1　离散事件仿真选项卡

Discrete Event Simulation（离散事件仿真）选项卡由三个平板组成：*Logs*（日志）、*Selected Log*（被选中的日志）和 *Messages*（消息）。*Logs* 平板包含所存在仿真日志的一个树形视图。默认情况下，这个平板仅包含当前场景的仿真日志。但是，如果您想的话，则可通过取消选择（unselecting）*Display logs for current scenarios only*（仅显示当前场景的日志）检查框，它位于 **Log Viewer** 窗口的底部。通过在一个仿真运行号上单击，可指定希望详细研究的 DES 日志消息的仿真运行。可一次仅详细研究单个仿真运行的 DES 日志。

Selected Log 平板使您可指定希望显示的日志消息的类型。默认情况下，没有选择日志类型，这样就会使当前仿真的所有 DES 日志消息都被显示。如果仿真运行产生大量 DES 日志消息，那么可以仅显示与当前正在详细研究的问题有关的那些消息，方法是单击期望的日志类。例如，可仅显示在网络中为一个特定节点产生的日志消息或与某个协议有关的日志消息；另外，可选择显示记录配置错误或告警等的那些消息。

Messages 平板包含与 DES 日志消息有关的信息。*Messages* 平板结构化为一个表的形式，其中每行包含有关单条 DES 日志消息的信息，而列代表诸如消息严重性（如通知、告警和错误）等的各种消息属性；当消息被记录时的仿真时间；导致产生 DES 日志消

息的事件号；消息为之产生的节点；DES 日志消息种类、类和子类；以及消息本身。在表中的 *Message* 字段上单击，打开一个 **Log Entry**（日志表项）窗口，包含消息的全描述，包括如下信息，如导致产生这条消息的仿真事件，这个事件对仿真的影响，对可能配置改变的建议（可能解决由这条 DES 日志消息描述的问题）。

在 **Log Viewer** 窗口底部，有四个控制项：

1）*Display logs for current scenarios only*（仅显示当前场景的日志）检查框，控制哪些 DES 日志将可用于 *Logs* 平板中的详细研究。

2）**Open Advanced Viewer**（打开高级查看器）按钮，当按下时，打开 **Advanced DES Log Viewer**（高级 DES 日志查看器）窗口，它为一个被选中仿真运行检查 DES 日志提供一个额外的选项。**Advanced DES Log Viewer** 使您可为搜索 DES 日志消息指定定制过滤器。在本书中不描述 **Advanced DES Log Viewer**。

3）**Close** 按钮，关闭 **Log Viewer** 窗口。

4）**Help** 按钮打开一个网页，提供 Log Viewer 窗口中可用选项的一个简短描述。

当打开 *Miscellaneous* 选项卡时，这些窗口控制保持不变。

4.8.2 杂项选项卡

如图 4.26 所示的 *Miscellaneous* 选项卡，仅有称作 *Logs* 和 *Selected Log* 的两个平板。通过 *Logs* 平板，可选择希望详细研究的两个可用日志文件中的哪个：Session log（会话日志）或 Error log（错误日志）。也可以通过分别打开 session_ log 或 err_ log ASCII 文本文件，来详细研究这些日志，这两个文件都位于 op_ admin 目录中。取决于您的选择，*Selected Log* 平板将显示会话日志或错误日志文件的内容。

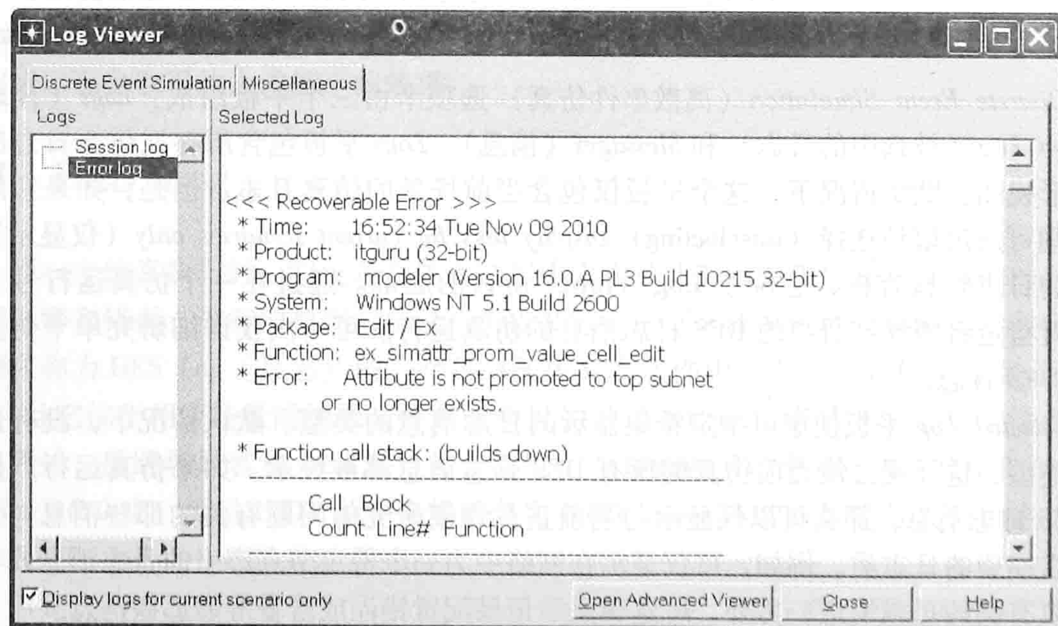


图 4.26 **Log Viewer** 窗口的杂项选项卡

第5章 标准应用

5.1 对OPNET中的流量源进行建模

流量产生来源形成网络系统建模中参数配置的一个关键方面。通过网络传输的数据流量可在各种网络协议的运行环境中观察它们的行为并研究它们的性能。在没有数据源的情况下，网络中仅有由某些协议和技术在初始化或周期性产生的控制报文（如在路由协议中的状态更新）。但是，这种控制流量总量是非常小的，且结果是，控制流量通常对整体网络性能具有非常小的影响（如果有影响的话）。因此，在多数情形中，为实施一个网络系统的研究，重要的是相应的仿真模型包括流量源，它们产生在被建模网络系统的各节点之间交换的数据报文。

OPNET提供可包括在一个仿真中的各种流量源模型。一个流量源如何部署在一个仿真模型中取决于模型中所需要的流量源的类型。一些类型的源被配置运行在个体网络节点上，并可以一种简单的方式进行创建，而其他类型的源则要求比较复杂的配置过程。

本章后面部分以及第6章、第7章描述在OPNET中流量源的配置和部署。5.2节开始解释OPNET中存在的流量源模型的不同类型。本章的其他各节仅讨论标准应用。5.3节描述了标准应用可被包括在一个OPNET仿真中的通用过程。5.4节提供了每个可用标准应用及其配置属性的描述。5.5节解释了当配置应用时如何使用网络节点的符号名（symbolic name）。5.6节描述可用于收集的标准应用统计量。第6章描述产生流量的其他高级OPNET特征，如显式的报文产生源、需求（demands）和基线负载。如果希望的话，可略过第6章，并直接阅读第7章，该章通过介绍用户概要（profiles）和在被仿真网络系统内部署标准和定制应用的方法，而结束在OPNET中对流量源建模的讨论。

5.2 在OPNET中流量源模型的类型

典型情况下，OPNET使用**explicit**（显式的）、**background**（背景）或**hybrid**（混合）（即显式的和背景的都有）流量模型来仿真网络流量。通过选择不同的机制，每种显式模型和背景模型都可部署在一个仿真中。图5.1形象地给出在OPNET中可用的流量模型的总体层次结构。在5.2.1～5.2.3节进一步描述这些模型。

5.2.1 显式流量模型

Explicit traffic models（显式流量模型）提供了流量行为的一个非常准确的描述，

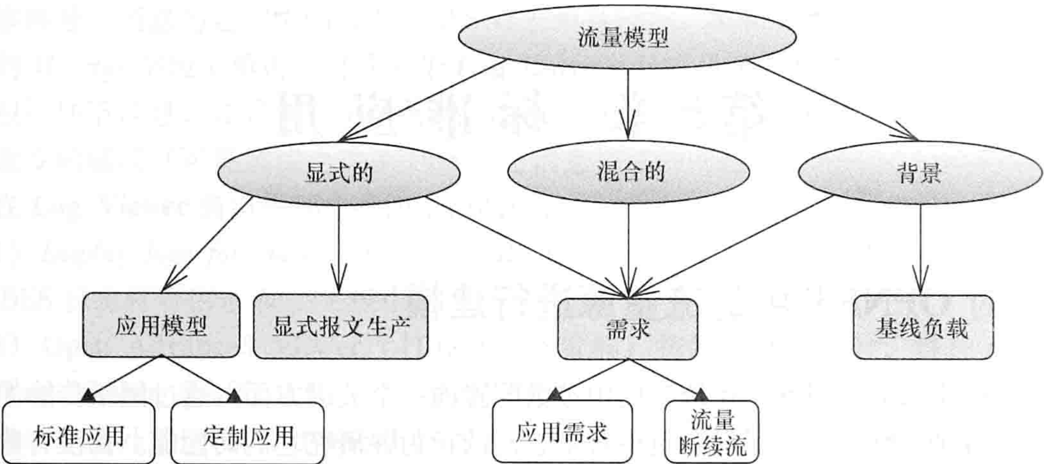


图 5.1 OPNET 的流量模型层次结构一览

通过对流量源产生的每条报文的完整生命周期建模，可做到这一点。报文流量是一个 OPNET 仿真的生命线，因为数据报文流过协议层才触发了那些协议中的各种功能。典型情况下，由流量源产生的数据被分割成消息或报文，之后被发送到下面的协议各层，它们进一步分段、封装、分片、排队、转发、接收、重新组装并实施其他处理，所有这些都是被仿真系统中各种协议的职责。在没有数据报文的情况下，仿真将不能对诸如 TCP、UDP、IP、资源预留协议（RSVP）、OSPF 等协议的所有特征进行精确的建模，因此，就难于准确地评估这些协议的性能。

但是，对网络流量的高精度逐报文建模是有代价的。显式的流量产生模型会导致高的内存消耗，因为每条报文都要被显式地建模，并在主存中表示为需要内存分配的一个数据结构。另外，显式的模型要求较长的仿真执行时间，因为每条报文的寿命周期的每个阶段都被仿真为一个或多个 DES 事件，每个事件都对执行时间有所贡献。

OPNET 为建模显式流量提供了如下机制：**explicit packet generation**（显式报文产生）、**application demands**（应用需求）和 **application models**（应用模型）。**explicit packet generation** 主要关注于指定流量被产生的速率、由个体流量源产生的报文的尺寸和流量目的地。这种方法不能对任意特定的应用层协议进行建模。使用显式报文产生来定义流量源的方法，要求修改产生流量的所有节点的属性值。在 OPNET 中，仅有某些节点模型［例如，典型的是流量源或目的（sink）节点］包含指定显式报文产生特征的属性。这样的节点模型通常在其名字中包含词_ station，并可通过搜索 **Object Palette Tree**（对象模板树）而容易地找到（见 2.3.2 节）。

application demand（应用需求）是指定一个被仿真网络系统中两个节点之间所交换显式流量的另一种机制。这种机制也主要关注于表示流量的速率和报文尺寸，它不对任何特定的协议行为建模。应用需求被实现为在两个节点的应用层之间交换的请求和响应消息的一个流。应用需求的配置不要求设置个体节点的属性值。相反，它仅涉及需求对象本身，它通过诸如流量不连续流的时长（即开始和结束时间）、请求和响应参数（即报文尺寸和流量速率）、传输协议和混合流量（traffic mix）等属性来定义两个节点

之间的流量不连续流。**Traffic Mix（%）**配置参数支持使用显式和背景流量模型对应用需求建模。特别地，这个属性将一个应用需求建模为背景流量的百分比。OPNET 也包含定义这种流量不连续流的其他需求模型，如话音、ATM 和 IP（通用的和特定的，如安全和 ping）。

application models（应用模型）也指定被仿真网络系统中节点之间交换的一条显式消息。但是，除了指定流量不连续流的特征（即速率、报文尺寸、目的地等）外，应用模型也关注表示应用层协议的行为或简单地说就是应用。这就是为什么提供被仿真系统中流量源的一种更准确表示，这比应用需求或显式报文产生要准确。OPNET 提供了标准应用的一个集合，已经实现对特定应用层协议的建模，这些协议如 FTP、HTTP、电子邮件、远程登录和其他协议，还为定义定制应用提供了一种机制，这可建模其他非标准应用层协议。

5.2.2　背景流量模型

Background traffic models（背景流量模型）提供了网络上数据传输的一种分析性的且因此不太准确的表示。在中间节点处队列的占用情况是基于背景流量模型的配置进行调节的，这顺次影响相应被仿真协议和网络设备的延迟和其他性能度量。背景流量模型不对个体报文的生命周期进行仿真；相反，它们采用流量行为的一种分析性表示。因此，包括背景流量模型的一项仿真研究将比使用显式流量模型的一项仿真研究，执行得更快并消耗较少的内存。

通过 **traffic demand objects**（流量需求对象）（即流量不连续流或应用需求）或 **baseline loads**（基线负载），对背景流量进行建模。**Demand objects**（需求对象）通常指定一对节点之间的流量。OPNET 提供了应用需求和其他需求类型间的差异，这被称作 **traffic flows**（流量不连续流）。应用需求以两节点之间交换的请求—响应消息的形式表示应用层流量（即流量是以在两个方向传输的方式进行配置和建模的）。流量不连续流被用来对其他非应用源产生的数据进行建模，这些源如 IP 话音、IP ping、IP 安全、IP 组播、ATM 和帧中继。流量不连续流表示通用的流量特征，诸如仅在一个方向传输的数据的速率、时长和开始时间。需求是以将相应的需求对象拖放到网络拓扑而添加到仿真的，之后配置它们表示所关注的流量不连续流。流量需求经常部署在没有直接连接到一条链路的节点之间。具体而言，需求不必与任何特定路径关联，在这种情形中，由仿真和被部署的路由协议确定流量如何通过网络进行传输。但是，需求配置的具体细节取决于正在处理的需求类型，将在 6.6 节讨论。注意任何流量需求可被配置以建模为纯粹离散的或显式的流量、纯粹背景流量或这两者的组合。

Baseline loads（基线负载），另外，仅代表应用到单个对象（如一条链路、节点或连接）的背景吞吐量。类似于显式报文生成，基线负载不要求任何附加对象被放置在项目工作空间内，可直接在所关注对象上进行配置。在本书中，仅讨论应用到链路模型的基线负载。典型情况下，基线负载指定有关通过那条特定链路的背景流量的有关信息，如平均报文尺寸（假定在整个仿真中保持恒定）和流量速率（可被配置为变化

的）。在链路上的总体流量负载是这样计算的，将穿过该链路的所有流量类型产生的负载求和，可包括显式流量、需求和基线负载。由基线负载引入的延迟影响通过相应链路的每条显式建模的报文。

5.2.3　混合流量模型

一般来说，显式流量模型适合于这样的仿真研究，它们要求对部署于被仿真网络中的协议和技术的详细建模，这接下来就使流量源行为的准确表示是必要的。另外，背景流量模型经常被用于这样的仿真研究，具有稀缺的计算资源，且其中协议的聚合行为足以用于研究的目的。在实践中，仿真研究经常依赖于显式模型和背景模型的一个组合，对被仿真网络系统中的流量源加以表示。这样的 **hybrid traffic models**（混合流量模型）将两种方法的优势组合起来，方法是通过使用显式模型并得到要求详细评估的那些流量源的准确表示，同时将背景模型用于不需要准确表示的流量源。结果是，混合模型提供优于背景模型的较大的准确性，同时相比显式模型，使用较少的资源并在执行上有所加速。

5.3　在一个仿真模型中包括应用

配置一个仿真系统来运行应用模型，是一个多步骤的过程，包括配置关注的应用和要在仿真过程中收集的相应统计量，指定用户概要（profile），并最终将配置的应用部署到被仿真网络系统中的节点上。OPNET 在一个 **application model**（应用模型）（指定由一个应用产生的流量的性质）和 **user profile**（用户概要）（描述应用将如何被使用）之间做出区分。一般而言，一个应用模型的定义由诸如报文间隔时间和尺寸等信息组成，而用户概要指定使用哪些应用，每个应用何时开始，每个应用执行多长时间，在概要内的应用是并行执行还是串行执行，整个概要何时开始和结束等。

在 OPNET 中，部署应用实际上意味着部署定义好的用户概要，这典型地包括确定概要将被在其上执行的节点和将为概要的应用提供服务的节点。运行用户概要的一个节点被称为支持用户概要，并被标记为一个 **source**（源）或一个 **client**（客户端）。类似地，为应用提供服务的一个节点被称作是支持应用的，并被标记为一个 **server**（服务器）或一个 **destination**（目的）。因此，客户端或源节点支持用户概要，而服务器或目的节点支持应用服务。

注意一个服务器可能并不支持所有的概要应用，因此，客户端节点需要联系不同服务器，以便得到为单个概要内应用提供的服务。进而，一个节点可同时作为客户端/源和作为服务器/目的地运行，这意味着同一节点可支持用户概要和应用。这样的一种设计是非常灵活的，且它支持密切反映真实网络中应用部署的应用配置。

考虑如下情形，其中一个网络仿真被配置三个用户概要。称它们为用户概要 1、2 和 3。每个这样的概要支持不同数量的应用。用户概要 1 代表仅允许电子邮件和打印应用的一名职员；概要 2 代表运行 Web、远程登录和电子邮件的一名学生；而概要 3 表示

仅接听电话，因此仅允许一个话音应用的一名顾客服务雇员。这些应用被部署在网络中，如表5.1所示。单个客户端节点（如节点A和C）可支持多个概要，也许代表在不同时间在该计算机上工作的不同用户或实施多项任务的一名用户或以不同角色进行操作的一名用户。类似地，单个服务器可支持多项应用（如节点D和E），这在今天的世界中是非常普遍的。最后，同一节点支持用户概要和应用（如节点F），也是可能的。

表5.1　应用和用户概要的例子

节点	所支持的用户概要	所支持的应用服务
节点 *A*	2和3	无
节点 *B*	2	无
节点 *C*	1和3	无
节点 *D*	无	电子邮件、Web（即HTTP）
节点 *E*	无	远程登录、电子邮件
节点 *F*	3	话音

5.3.1　Application Config 设施对象

指定和配置标准应用的第一步是在被仿真系统中包括一个 *Application Config*（应用配置）设施。通常是采取如下方法做到这一点的，打开 **Object Palette Tree**（见2.3节），之后选择 *Application Config* 对象并将其拖放到项目工作空间。*Application Config* 对象的图标如图5.2所示。典型地，可在如下 **Object Palette Tree** 类中找到 *Application Config* 对象：



图5.2　*Application Config* 设施对象

1）*Node Models→Fixed Node Models→By Name→Application*（节点模型→固定节点模型→依名字→应用）。

2）*Node Models→Fixed Node Models→By Machine Type→utility*（节点模型→固定节点模型→依机器类型→设施）。

3）*Shared Object Palettes→applications→Node Models*（共享的对象调色板→应用→节点模型）。

4）*Shared Object Palettes→internet_ toolbox→Node Models*（共享的对象调色板→互联网工具箱→节点模型）。

5）*Shared Object Palettes→utilities→ Node Models*（共享的对象调色板→设施→节点模型）。

6）其他方式。

一个被仿真的场景不应该包含多个 *Application Config* 对象。为配置所有必要的标准和定制应用，仅需要一个 *Application Config* 对象。图5.3给出 *Application Config* 对象的 **Attributes**（属性）窗口。正如您会想到的，这个窗口可通过在 *Application Config* 对象

上右击，并选择 **Edit Attributes** 选项的方式来打开。

如图 5.3 所示，*Application Config* 对象包含分别称作 **Application Definitions**（应用定义）、**MOS** 和 **Voice Encoder Schemes**（话音编码器方案）的三个复合属性。后两个属性处理话音应用的建模，而我们仅关心 **Application Definitions** 复合属性，它定义标准应用和定制应用的配置。

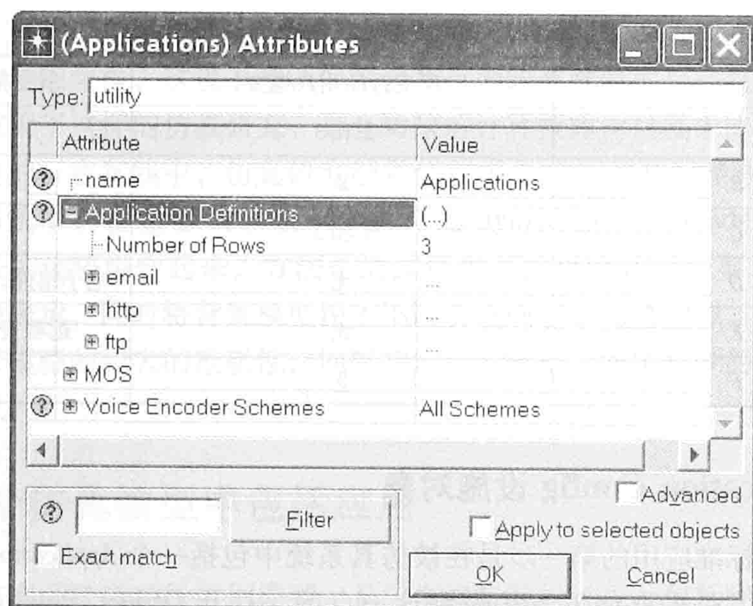


图 5.3 *Application Config* 对象的 **Attributes** 窗口

Application Definitions 复合属性包含属性 **Number of Rows**（行数），确定在 *Application Config* 对象内要配置的应用数量。**Number of Rows** 的默认值为 0，表示在当前场景中没有配置应用。将 **Number of Rows** 设置为大于 0 的任何值，如 A，导致在 *Application Definitions* 复合属性内创建 A 个表项，每个表项负责提供一个标准应用或定制应用的单一定义。例如，图 5.4a 显示，**Number of Rows** 被设置为 3，这得到三个应用定义表项。因为还没有配置任何表项，所以每个表项被命名为 Enter Application Name...（输入应用名字...）。

一个应用定义表项有一个属性 **Name**（指定当前应用的一个字母数字组成的名字）和一个复合属性 **Description**（由 10 个复合属性组成，为 1 个定制应用定义和 9 个标准应用定义指定参数）。仅有这些复合参数之一可为每个单一应用定义表项进行配置。因此，即使属性 **Description** 包含 10 个应用配置属性，但仅有这些属性之一可被配置，而其他的属性必须保持设置为值 Off（关闭）。图 5.4b 给出图 5.4a 中相同的对象，但前两行都被设置为模型 Database 应用。

当配置 OPNET 应用时，新手用户普遍会犯如下错误之一：

1）在项目工作空间内放置多个 *Application Config* 对象，为每个应用定义放置一个。这是不正确的，因为单个 *Application Config* 对象使您可指定多个应用定义。

2）通过在复合属性 **Description** 之下设置几个应用定义，尝试在单个应用定义内配

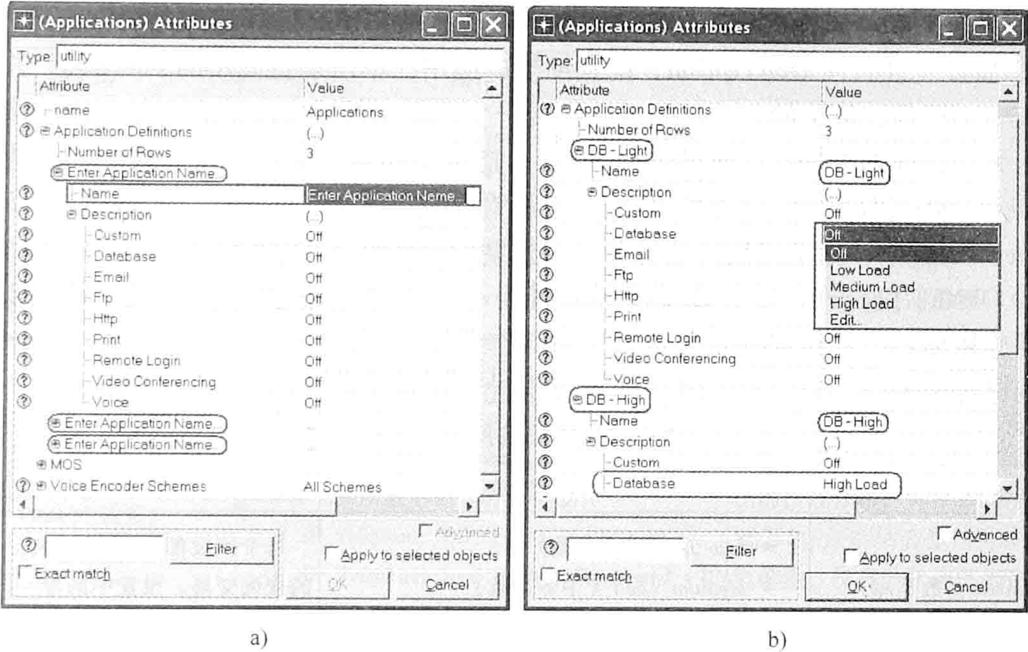


图5.4　配置 **Application Definitions** 复合属性

a）有三行，都还没有进行配置　b）前两行都配置为对数据库应用建模

置多项应用。这种尝试将导致一条 OPNET 告警出现在 **Application Attributes** 窗口顶部或底部，声称 "Only one application type can be set per application name"（每个应用名称只能设置一个应用类型）。

另外，所有应用都运行在一个特定的默认传输协议上。HTTP、FTP、E－mail、打印、数据库和远程登录应用，默认情况下都运行在 TCP 之上，而视频会议和话音应用，默认情况下都运行在 UDP 之上。定制应用的定义包含一个属性，显式地配置由那个定制应用采用的传输协议（见6.2节）。但是，如果需要，任何标准应用和定制应用都可在客户端节点和服务器节点处改变它们的默认传输协议。在7.5.5节讨论做出这样一个改变的过程。

5.3.2　配置标准应用

在5.4节详细描述 OPNET 中可用的标准应用。那个小节也提供了为这些应用的每个应用可能需要配置的属性的一个解释。在一个 OPNET 仿真中配置标准应用的通用指令如下：

1）添加 *Application Config* 对象（如果它还不在项目工作空间中的话）。

2）在 *Application Config* 对象上右击，并选择 **Edit Attributes**。

3）展开 **Application Definitions** 复合属性。

4）指定 **Number of Rows**（行数）属性的值，它对应于在当前仿真场景中要配置的不同应用总数。

5）展开各行，每次展开一个来配置每项应用。

① 设置 **Name** 属性的值，这将自动地改变包含应用定义的相应行的名字。

② 展开复合属性 **Description**，配置期望的应用：单击期望应用的 *Value* 列，并从下拉列表中选择预设值中的一个。表 5.2 给出一个列表和预设应用值之间差异的一个简短描述。另外一种方法是，可通过选择 Edit... 指定个体属性的值，这会打开相应应用的 *Attributes Table*（属性表）。也可通过双击期望应用的 *Value* 列访问 *Attributes Table*。在 *Attributes Table* 中，指定应用属性的值，与预期的一样来定义应用。

③ OPNET 仅允许在单个 **Description** 属性（即每行）内配置一个应用，而在那个属性内的应用描述的其他部分必须被设置为 Off（关闭）值。事实上，如果在 **Description** 属性内已经配置另一项应用，OPNET 就不会允许改变任何行中的 Off 设置。

表 5.2 预设应用值的描述

应用	预设值	简短描述
Database（数据库）	• *Low Load*（低负载） • *Medium Load*（中等负载） • *High Load*（高负载）	每个预设值被配置为执行 100% 的查询交易。设置中的唯一差异是交易响应的尺寸和交易到达的频率
E-mail（电子邮件）	• *Low Load*（低负载） • *Medium Load*（中等负载） • *High Load*（高负载）	每个预设值被配置为每个组发送和接收三条电子邮件消息。在配置中的主要差异是电子邮件消息的尺寸和发送与接收消息的间隔时间
FTP	• *Low Load*（低负载） • *Medium Load*（中等负载） • *High Load*（高负载）	每个预设值被配置为使所有 FTP 操作均匀地分布在 get 和 put 操作之间。配置中的主要差异是被传输文件的尺寸和 FTP 操作产生的频率
HTTP	• *Light Browsing*（轻度浏览） • *Heavy Browsing*（重度浏览） • *Searching*（搜索） • *Image Browsing*（图像浏览）	每个值被配置为运行 HTTP 1.1。配置中的主要差异是页面请求的频率、网页的定义和用户的网页浏览行为的确定
Print（打印）	• *Text File*（文本文件） • *B/W Images*（B/W 图像） • *Color Prints*（彩色打印）	这些预设属性在打印任务的频率和文件尺寸方面存在差异，*Text File* 设置具有最小的文件尺寸和最高的频率，而 *Color Prints* 具有最大的尺寸和最低的任务提交频率
Remote Login（远程登录）	• *Low Load*（低负载） • *Medium Load*（中等负载） • *High Load*（高负载）	每个预设值被配置为具有主机和终端命令的不同产生频率和尺寸

（续）

应用	预设值	简短描述
Video Conferencing（视频会议）	• *Low – Resolution Video*（低分辨率视频） • *High – Resolution Video*（高分辨率视频） • *VCR Quality Video*（VCR 质量视频）	在这些设置之间的唯一差异是视频帧的尺寸及其产生的频率
Voice（话音）	• *PCM Quality Speech*（PCM 质量讲话） • *PCM Quality Speech and Silence Suppressed*（PCM 质量讲话和静音抑制） • *Low – Quality Speech*（低质量讲话） • *Low – Quality Speech and Silence Suppressed*（低质量讲话和静音抑制） • *IP Telephony*（IP 电话） • *IP Telephony and Silence Suppressed*（IP 电话和静音抑制） • *GSM Quality Speech*（GSM 质量讲话） • *GSM Quality Speech and Silence Suppressed*（GSM 质量讲话和静音抑制）	这些预设值之间的唯一差异是所用话音编码方案的类型和 ToS 字段的值。属性值的其他部分是相同的：静音和讲话突发的长度分别被设置为具有 0.65 和 0.352s 均值输出的指数分布；每秒话音帧数被设置为 1，所有话音数据被建模为离散流量，没有设置会使用任何信令协议，压缩和解压缩延迟被设置为 0.02s，话音应用客户端被配置为通过一条地面线路、位于一个安静房间的一次呼叫发生的

5.4　标准应用的描述

所有标准应用都对 **two – tier**（两层）、请求—响应应用层协议建模，其中一个客户端节点向一个服务器发出一条请求，那个服务器向那个客户端发送回一条直接响应。定制应用支持对具有两层以上的应用进行建模，其中一条客户端请求，在一个响应消息被发回之前，可通过多个服务器进行转发。我们将称这样的应用为 **multi – tier**（多层）。图 5.5 形象地说明了两层和多层应用之间的区别。

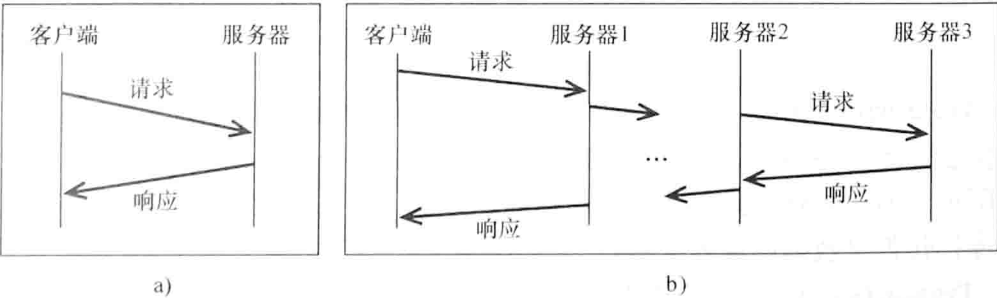


图 5.5

a）两层应用的例子　b）多层应用的例子

OPNET 中的标准应用包括 Database（数据库）、E – mail（电子邮件）、FTP、HTTP、Print（打印）、Remote Login（远程登录）、Video Conferencing（视频会议）和 Voice

(话音)。5.4.1~5.4.8 节描述这些应用以及它们的配置属性。

5.4.1 数据库

数据库应用被建模为执行两种数据库操作类型的一个协议：查询（query）和录入(entry)。**database query**（数据库查询）操作从数据库中检索数据。它由一条查询消息（携带数据库请求）和一条响应消息（携带数据）组成。对于查询操作，查询消息的尺寸总是512 字节，而数据库响应的尺寸是通过 **Transaction Size**（交易尺寸）属性可配置的。**database entry**（数据库输入）操作将数据写入数据库。它由携带数据的一条输入消息和携带操作的数据库确认的一条响应消息组成。对于数据库输入操作，一条输入消息（携带数据）的尺寸也是通过 **Transaction Size**（交易尺寸）可配置的，而一条响应消息的尺寸总是被设置为 512 字节。

默认情况下，数据库应用运行在 TCP 之上。当配置仿真在 TCP 上运行数据库应用时，那么在客户端节点和数据库服务器之间建立单条 TCP 连接，且这条连接被用于实施所有的数据库查询和输入交易。Database Configuration Table（数据库配置表）包含 7 个属性，如图 5.6 所示。前三个属性是特定于数据库应用的，而后四个属性是通用于许多其他标准应用定义的。后四个属性只在本节中进行介绍，后文不再重复介绍。

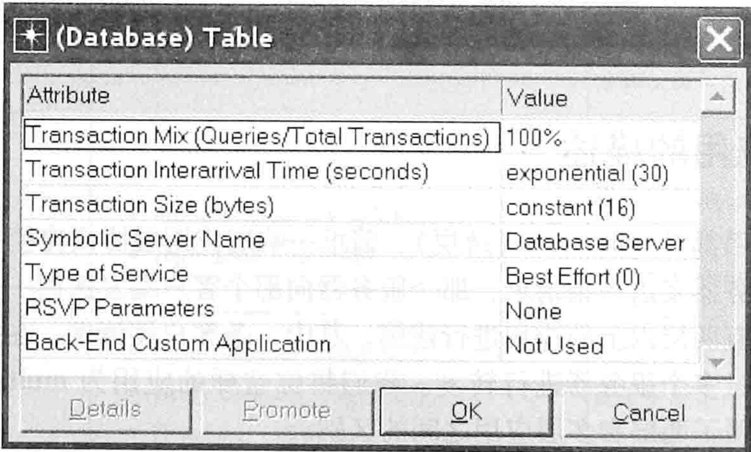


图 5.6 数据库配置表

1）**Transaction Mix**（**Queries/Total Transactions**）[交易混合型（查询数/总交易数）] 指定在所有数据库操作中查询交易的百分比。如果这个属性值设置为 100%，如图 5.6 所示，那么所有数据库交易都是查询。另外，如果属性值被设置为 0%，那么所有数据库操作都是数据库输入交易。

2）**Transaction Interarrival Time**（**seconds**）[交易到达间隔时间（秒）] 指定两次连续数据库交易之间时间间隔的时长。这个属性有助于确定下一次数据库交易的开始时间。下一次交易的开始时间是这样计算的，将这个属性的值加到前一次交易开始的时间。数据库交易是相互独立的，这意味着在接收到前一条交易的响应之前，客户端可开始一条新的交易。

3）**Transaction Size**（**bytes**）［交易尺寸（字节）］以字节为单位定义数据库交易请求的尺寸。对于数据库输入交易，这个属性定义请求消息的尺寸，而响应的尺寸被设置为512字节。对于数据库查询交易，情况相反：请求的尺寸被设置为512字节，而响应的尺寸被设置为由这个属性定义的值。

4）**Symbolic Server Name**（符号服务器名）为作为一给定应用定义的数据库服务器运行的所有节点，指定一个共同的符号名（即同一应用的不同定义可能具有不同的符号服务器名）。这个属性是每个标准应用定义的组成部分（定制应用除外）。在应用部署过程中，由这个属性定义的符号服务器名，被映射到仿真中物理节点的实际名字。符号服务器名是所谓目的首选定义的组成部分，它指定支持这项应用的服务器。具体而言，目的首选指定客户端将联系哪些服务器（即客户端可配置为仅联系支持这项应用的被选择的服务器）和被选服务器将被联系的概率（即客户端将被配置为以不同概率连续不同服务器）。在5.5节将以大量细节讨论这个话题。典型情况下，这个属性的值保持不变。

5）**Type of Service**（服务类型）是被指派到IP首部中服务类型（ToS）字节或区分服务码点（DSCP）字段的值，该IP首部是指由这项应用的客户端产生的每个个体报文中的IP首部。当以各种服务质量（QoS）机制一起被采用时，这个值指定在IP队列中报文的类型、类或“优先级”，并确定当报文穿过网络时它将如何被处理。典型情况下，报文处理机制是通过*IP QoS Attrib*（IP QoS属性）对象进行配置的。在第10章详细讨论这个问题。这个属性是多数应用定义的组成部分，其中包括定制应用，但HTTP应用不在其列。在实践中，这个属性的值经常保持不变，原因是它对仿真没有影响，除非仿真系统被配置为支持QoS机制，该机制依赖ToS或DSCP标记才能运行。

6）**RSVP Parameters**（RSVP参数）是一个复合属性，配置资源预留协议（RSVP），方法是确定这样的子属性，如**RSVP Status**（RSVP状态）（即这项应用是否支持RSVP）、**Outbound Flow**（外发流）（为这项应用预留的带宽和缓冲尺寸）和**Inbound Flow**（进入的流）（为对等端应用的流量预留的带宽和缓冲尺寸）。必须首先在*IP QoS Attrib*对象中定义流特征参数的值，仅有此时它们才可用于设置**RSVP Parameters**。这个属性是多数应用定义的组成部分，其中包括定制应用，但Print应用排除在外。在实践中，这个属性的值经常保持设置为None（无），原因是它对仿真没有影响，除非被仿真的系统配置为支持RSVP。

7）**Back – End Custom Application**（后端定制应用）指定定制应用的名字和这个定制应用的概率（在到达客户端的请求时，在应用服务器上执行该应用）。如果配置这个属性，那么在客户端的请求到达时，服务器将执行指定的定制应用。仅当服务器完成定制应用的执行时，它才将一条响应发回客户端。这个属性支持将标准两层应用与多层定制应用组合起来。这个属性是多数应用定义的组成部分，但HTTP、Print、Video Conferencing（视频会议）和Voice（话音）排除在外。在实践中，这个属性的值经常保持设置为Not Used（未使用），除非被仿真的系统要求在另一个应用的后端执行一个多层定制应用。

5.4.2　电子邮件

OPNET 将电子邮件应用建模为在一个电子邮件客户端和一个电子邮件服务器之间的两层请求—响应消息交换。电子邮件响应和请求操作是客户端驱动的：电子邮件客户端周期性地查询服务器来检索它的电子邮件消息，且它也可周期性地将新撰写完成的电子邮件发送给服务器。默认情况下，电子邮件应用使用 TCP 作为它的传输协议。当一个电子邮件应用被配置为在 TCP 之上运行时，那么在客户端和服务器之间打开单条 TCP 连接，且所有消息都通过那条 TCP 连接进行发送和接收。OPNET 没有对客户端到客户端的电子邮件消息交换进行建模。开发一个多层应用模型，其中一个客户端节点向一个服务器发送消息，之后那条消息由另一个客户端从服务器检索，这要求使用定制应用。

E-mail Configuration Table（电子邮件配置表）包含 9 个属性。但是，我们仅描述定义电子邮件的 5 个关键属性（见图 5.7），而其他 4 个属性 **Symbolic Server Name**（符号服务器名）、**Type of Service**（服务类型）、**RSVP Parameters**（RSVP 参数）和 **Back-End Custom Application**（后端定制应用），为避免重复，这里不再重述。

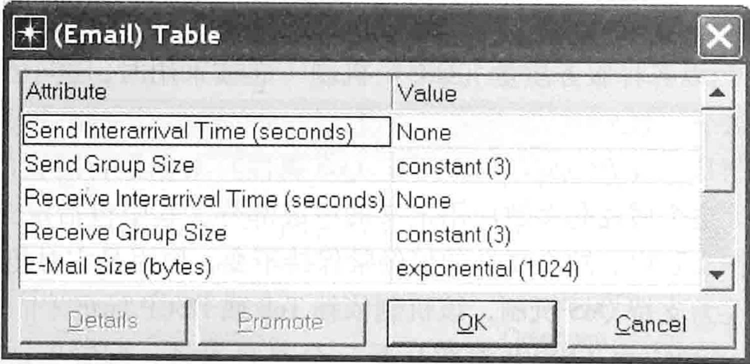


图 5.7　电子邮件配置表

1）**Send Interarrival Time（seconds）**［发送间隔时间（秒）］指定电子邮件下一批次将被上传到电子邮件服务器的时间。下一电子邮件批次的发送时间的计算：将这个属性的值加到前一电子邮件上传的时间。发送和接收到达间隔时间是相互独立的。这意味着客户端可被配置为频繁地将电子邮件消息上传到服务器，同时很少查询服务器来检索其电子邮件，反之亦然。

2）**Send Group Size**（发送组尺寸）指定要上传的每个批次电子邮件的消息数。

3）**Receive Interarrival Time（seconds）**［检索到达间隔时间（秒）］指定电子邮件消息的下一批次将从电子邮件服务器接收的时间。下一电子邮件批次的接收时间的计算：将这个属性的值加到前一批次电子邮件从服务器到达的时间。

4）**Receive Group Size**（接收组尺寸）指定要接收的电子邮件每个批次电子邮件消息数。

5）**E-mail Size（bytes）**［电子邮件尺寸（字节）］以字节表示的单个电子邮件消息的尺寸。如果客户端在每个电子邮件批次上传 5 条消息，且如果这个属性是 100 字节

（即每个电子邮件的尺寸为 100 字节），那么在每次上传中由客户端节点发送到服务器的数据总量是 5 × 100 字节 = 500 字节。

5.4.3　FTP

FTP 标准应用对文件传输协议（FTP）的基本操作进行了建模。即使常规 FTP 应用由多条命令组成，但这个 OPNET 模型也仅仿真数据传输的两个主要 FTP 操作：put 和 get。**FTP put** 操作将一个文件上传到 FTP 服务器，而 **FTP get** 操作将一个文件从 FTP 服务器下载到客户端节点。这两个操作由两种消息类型组成：控制和数据。**Control messages**（控制消息）是对一个文件的请求（即 get 操作）或对一个文件传输完成的确认（即在 put 操作中）。**Data messages**（数据消息）携带正在客户端和服务器之间传输的一个文件。控制消息尺寸总是 512 字节，而数据消息的尺寸是可配置的。

在 OPNET 中，FTP 应用仅对一次一个文件传输进行建模。默认情况下，FTP 运行在 TCP 之上。当使用默认传输协议，为每个文件传输操作打开一个独立的 TCP 连接。不像在真实网络中的是，这个 FTP 模型在同一 TCP 连接上一个文件传输操作的发送控制和数据消息进行建模。但是，仅描述定义 FTP 的三个关键属性，如图 5.8 所示。

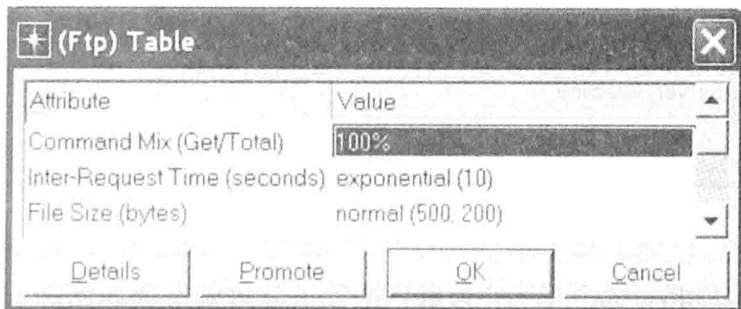


图 5.8　FTP 配置表

1）**Command Mix（Get/Total）**［命令混合型（获取/总量）］指定 get 操作数和 FTP 操作总数之间的比率。例如，如果这个属性的值被设置为 40%，那么所有 FTP 操作的 40% 将是 get 操作，而剩下的 60% 将是 put 操作。

2）**Inter – Request Time（seconds）**［请求间隔时间（秒）］指定连续 FTP 操作之间的时间总量。下一文件传输的开始时间如下：将这个属性的值加到前一 FTP 操作开始时的时间上。FTP 操作是相互独立的，这意味着一次新的文件传输可在前一 FTP 操作完成之前开始。

3）**File Size（bytes）**［文件尺寸（字节）］以字节为单位指定要传输的文件尺寸。

5.4.4　HTTP

OPNET 的 HTTP 应用仿真互联网浏览活动，其中一个客户端节点周期性地联系 Web 服务器，检索网页。在 OPNET 中，每个网页被建模为文本（一个基本的 HTML 文件）和几个内联对象（即诸如图像和数据文件的被引用对象）的一个组合体。超文本传输

协议（HTTP）使用TCP作为它的传输协议，其运行过程如下：客户端发送对一个网页的一条HTTP请求；服务器接收请求，并将相应的网页发回客户端。在这个网页的剖析过程中，如果该网页包含几个内联对象，那么客户端节点后续地从服务器请求这些对象。

HTTP版本确定各对象是如何在网络上传输的。例如，在HTTP版本1.0中，客户端节点使用一个非永久的连接，其中每个网页对象的请求和传输是在一个独立的TCP连接上进行的。另外，HTTP 1.1采用永久连接，其中网页对象是在同一条TCP连接上传输的。HTTP 1.1也使用流水线技术，其中将对各内联对象的多个请求组合为单条消息，这一点不像HTTP 1.0那样独立地发送每条对象请求。

HTTP Configuration Table（HTTP配置表）有描述HTTP应用的四个关键属性（见图5.9）及两个附加属性 **RSVP Parameters** 和 **Type of Service**（对许多应用定义是通用的，在5.4.1节描述）。下面给出HTTP应用属性的一个描述。

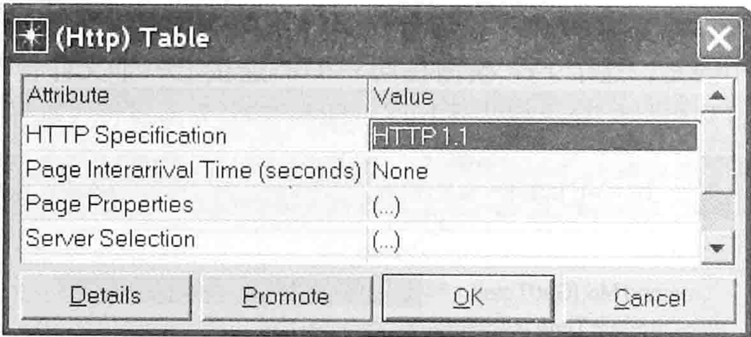


图5.9　HTTP配置表

1）**HTTP Specification**（HTTP规格）定义由网页浏览器使用的HTTP的配置。如图5.10所示，这个复合属性由如下子属性组成：

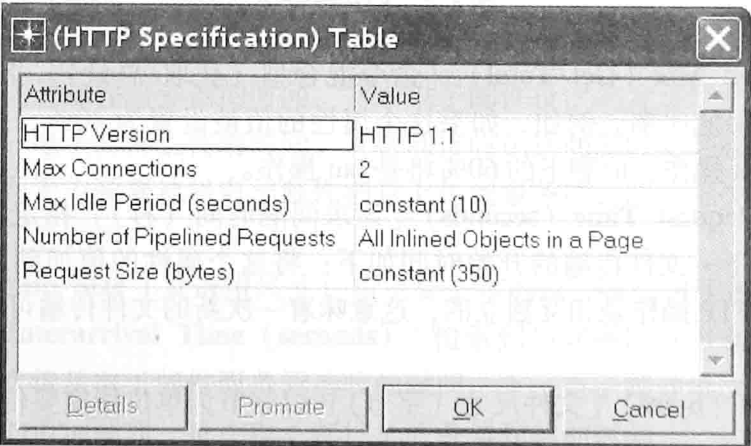


图5.10　HTTP规格复合属性

① **HTTP Version**（HTTP版本）定义HTTP的版本，有两个可能值：HTTP 1.0（支持非永久连接，且不采用流水线）和HTTP 1.1（支持带有流水线的永久连接）。

② **Max Connections**（最大连接数）为可在客户端和服务器之间建立的并行连接的最大数量。这个属性仅适用于 HTTP 版本 1.0，支持为网页对象请求传输而打开多条并行连接。注意，并行连接数不同于 HTTP 1.1 中的流水线，后者中多个网页对象请求是被组合到单条消息中的，并在单条连接之上进行传输。

③ **Max Idle Period（seconds）**［最大空闲周期（秒）］属性确定在一条 HTTP 连接超时之前它可空闲多长时间。这个属性仅适用于 HTTP 1.1，在有其他页面或对象要在同一连接上检索的情形中，这可保持一条连接处于打开的状态。

④ **Number of Pipelined Requests**（流水线请求的数量）指定可以流水线打包（即组合在一起）到单一传输层协议消息中的请求数量。这个属性仅适用于 HTTP 1.1。

⑤ **Request Size（bytes）**［请求尺寸（字节）］指定一个网页请求的尺寸。

2）为了简化 Web 应用的配置，OPNET 为 **HTTP Specification** 属性提供四个预设值。这些预设值代表 HTTP 的不同配置。每个这样的预设值将请求尺寸定义为 350 字节，而其他属性如下设置：

① HTTP 1.0——被配置为运行 HTTP 1.0，最大并行连接数设置为 4。

② HTTP 1.1——被配置为运行 HTTP 1.1，将最大空闲周期设置为 10s，并在网页内支持所有内联对象的流水线处理。

③ Microsoft IE 5.0——被配置为运行 HTTP 1.1，将最大空闲周期设置为 10s，且不支持流水线处理。

④ Netscape Communicator 4.0——被配置为运行 HTTP 1.0，最大并行连接数设置为 4。

3）**Page Interarrival Time（seconds）**［页面间隔时间（秒）］指定连续网页请求之间的时间。这个属性实际上定义了一名用户的网页浏览行为。例如，如果希望对从网上阅读新闻文章的一名用户进行建模，那么这个属性应该设置为一个相对大的值。如果假定一个常人阅读一篇文章的时间约为 5min，那么应该将这个属性的值设置为 300s。另外，如果希望对搜索网络以便得到信息的用户进行建模，那么这个属性的值应该要小得多；假定是 10s，这大约是一名用户在搜索时查看一个网页要花费的平均时间。注意这个属性是通过一个概率分布函数定义的，这就使建模随机变化的值并由此建模一个更真实的用户行为成为可能。

4）**Page Properties**（页面性质）描述一个典型网页的一个 OPNET 表示。这个属性指定信息，如基本 HTML 网络（即首页）的尺寸、内联对象的数量和尺寸。每个页面对象被表示为表中的一行，如图 5.11 所示。下面给出 **Page Properties** 复合属性表中各列的描述：

① **Object Size（bytes）**［对象尺寸（字节）］指定网页内单个对象的尺寸。

② **Number of Objects（objects per page）**［对象数量（每页的对象数）］指定网页内某个类型的对象数。表中的第一行总是表示基本 HTML 文件。因为每个网页仅可有一个基本 HTML 文件，所以第一行中的对象数总是 1，而不管 **Number of Objects**（对象数）列中设置的实际值为何。

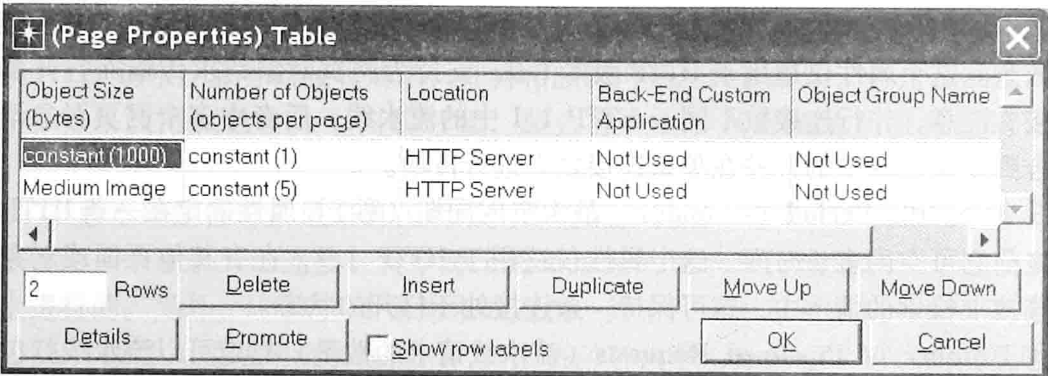


图 5.11　Page Properties 复合属性

③ **Location**（位置）指定 HTTP 服务器（内联网对象驻留其上）的符号名。即使内联对象可有相同的符号服务器名，物理上它们也可能位于不同节点上，因此客户端就需要联系不同服务器来检索单个网络的所有内联对象。否则，这个属性具有 **Symbolic Server Name**（符号服务器名）属性相同的含义（见 5.4.1 节）。

④ **Back – End Custom Application**（后端定制应用）定义要在服务器的后端使用的定制应用（见 5.4.1 节）。

⑤ **Object Group Name**（对象组名）指定具有这些特征的网页对象的一个常用名。这个属性用于收集 *Object Response*（对象响应）统计量，其中是基于对象组的名字而聚集仿真结果的。

5）**Server Selection**（服务器选择）定义通过内嵌网页链接索引的网页是将从相同服务器还是从另一个服务器进行检索。网页经常包含指向其他网页的内嵌链接。这个属性指定指向网页的内嵌链接是否位于同一服务器上。具体而言，这个复合属性由如图 5.12 所示的两个子属性组成：

① **Initial Repeat Probability**（初始重复概率）指定内嵌于网页（第一次被访问）的链接位于同一服务器上的概率。这个属性的值范围从 0.0 到 1.0（包括边界）。

② **Pages Per Server**（每个服务器的页数）指定连续从同一服务器访问的网页数。

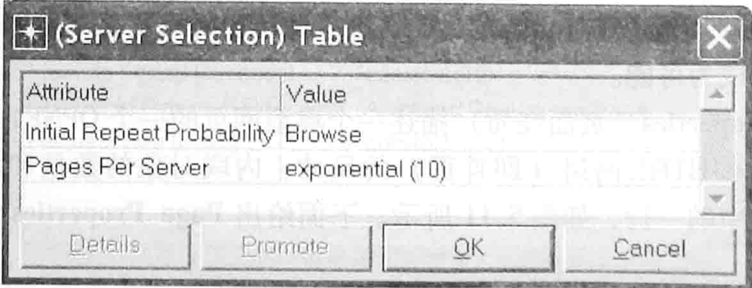


图 5.12　Server Selection 复合属性

总之，这些属性对如下服务器选择行为进行了建模。如果网页是从当前服务器检索的第一个网页，那么仿真使用 **Initial Repeat Probablity** 属性的值来确定这个服务器是否再次会被访问。如果选择同一服务器，那么客户端从那个服务器检索 *N* 个连续的网

页，其中 N 是由 **Pages Per Server** 属性指定的值。如果仿真确定应该选择一个不同的服务器或如果所有 N 个网页已经从当前服务器访问了，那么基于客户端的目的首选项，选择一个新的服务器。OPNET 为 **Initial Repeat Probability** 属性提供三个预设值，如此命名是为了代表各种用户行为：

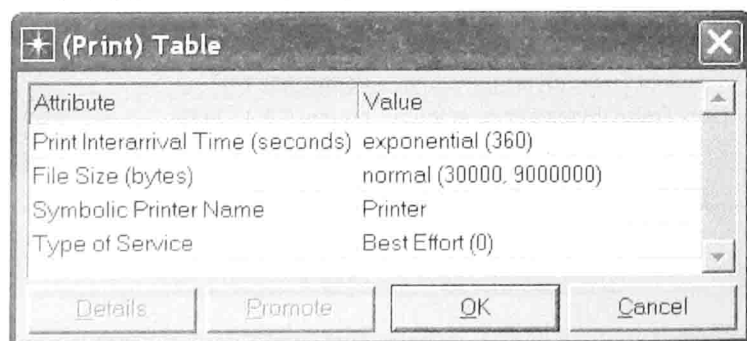
1）Searching（搜索）。当搜索 Web 时，典型情况下用户访问的各种网页经常位于不同站点。而且，用户将经常每个站点访问单个页面。这两个因素意味着，下一网页位于同一服务器上的概率应该较小。那就是为什么表示这样一种行为，**Initial Repeat Probability** 属性的值设置为 0.3。

2）Browsing（浏览）。当浏览 Web 时，用户访问多个站点，但在移到一个不同的站点之前经常探索同一服务器上的多条链接。为了表示这样的行为，**Initial Repeat Probability** 属性的值被设置为 0.5。

3）Research（研究）。当做研究时，典型情况下，用户访问来自同一站点的多个网页，但仅访问少量不同的 Web 服务器。这就是为什么为建模这样的行为将 **Initial Repeat Probability** 属性的值设置为 0.9。

5.4.5　打印

Print 应用对提交一项打印任务到一个打印服务器或一个打印机的操作进行建模。OPNET 甚至具有表示网络打印机的几个节点模型。打印应用的目的地可以是一个打印机或一个打印服务器。打印应用运行在 TCP 之上，并为每个打印任务请求发起一条新的 TCP 连接。*Print Configuration Table*（打印配置表）由四个属性组成，如图 5.13 所示。这里不会提供对 **Symbolic Printer Name**（符号打印机名）和 **Type of Service** 属性的描述，原因是其中之一（**Type of Service**）在 5.4.1 节做了描述，而另一个（**Symbolic Printer Name**）类似于 5.4.1 节中描述的属性 **Symbolic Server Name**。


(Print) Table

Attribute	Value
Print Interarrival Time (seconds)	exponential (360)
File Size (bytes)	normal (30000, 9000000)
Symbolic Printer Name	Printer
Type of Service	Best Effort (0)

图 5.13　打印配置表

1）**Print Interarrival Time（seconds）**［打印间隔时间（秒）］指定两次连续打印任务请求之间消逝的时间。自前一打印任务请求发起过去“interarrival time”（间隔时间）秒之后，开始下一个打印任务。

2）**File Size（bytes）**［文件尺寸（字节）］以字节为单位指定被转发到一个打印服务器或一个打印机的单个打印任务的尺寸。

5.4.6 远程登录

远程登录应用对一项虚拟终端服务（也称为一个终端网络或 Telnet）进行建模。远程登录应用使用户能够连接到一个远端服务器，并通过从一台本地机器上发出命令在服务器上实施各种操作。发自一个本地系统的命令和由远端服务器产生的响应一起，产生了通过本地节点和远端节点之间的网络传输的流量。

默认情况下，远程登录应用运行在 TCP 之上，且每个远程登录会话都要求一条独立的 TCP 连接。OPNET 称客户端节点接收到的流量为 *Host Traffic*（主机流量），称由客户端节点发送的流量（即由远端终端接收到的）为 *Terminal Traffic*（终端流量）。*Remote Login Configuration Table*（远程登录配置表）总共包含 7 个属性。但是，这里仅描述如图 5.14 所示定义远程登录应用的三个关键属性。为避免重复，略去其他 4 个属性 **Symbolic Server Name**（符号服务器名）、**Type of Service**（服务类型）、**RSVP Parameters**（RSVP 参数）和 **Back - End Custom Application**（后端定制应用），原因是在 5.4.1 节对它们做了描述。

1）**Inter - Command Time**（**seconds**）［命令间时间（秒）］指定两条连续远程登录命令之间的时间。从前一条命令被执行开始过去“intercommand time”（命令间时间）秒之后，发出下一条命令。

2）**Terminal Traffic**（**bytes per command**）［终端流量（每条命令的字节数）］以字节为单位，定义由本地主机产生的被发送到远端目的地的每条命令的尺寸。

3）**Host Traffic**（**bytes per command**）［主机流量（每条命令的字节数）］以字节为单位定义由本地主机从远端目的地接收到的每个响应的尺寸。

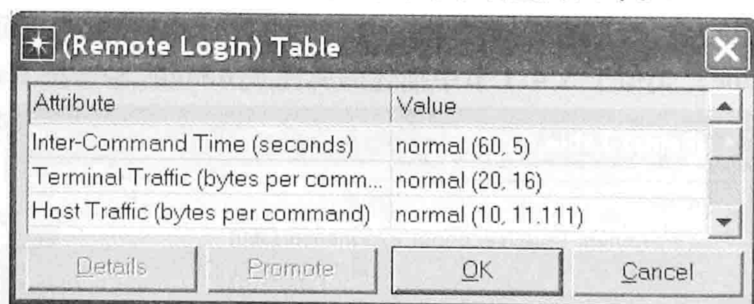


图 5.14 远程登录配置表

5.4.7 视频会议

视频会议应用对网络中两个节点之间视频流量的传输进行建模。OPNET 将视频流量表示为帧尺寸是一个可配置参数的数据帧的一个序列。默认情况下，视频会议应用是运行在 UDP 之上的，目的是避免与 TCP 相关联的连接管理和其他延迟。但是，如果视频会议配置为运行在 TCP 之上，那么每个客户端为一条单向的视频连续流打开一条独立的连接。典型情况下，一个视频会议会话是在不使用一台服务器的条件下，在两个客户端节点之间建立的。

Video Conferencing Configuration Table（视频会议配置表）包含总共 6 个属性，但是，这里仅描述两个关键属性，如图 5.15 所示。这些属性指定由视频会议应用产生的流量负载的特点。为了避免重复，略去其他 4 个属性（**Symbolic Server Name**、**Type of Service**、**RSVP Parameters** 和 **Back – End Custom Application**）的描述。

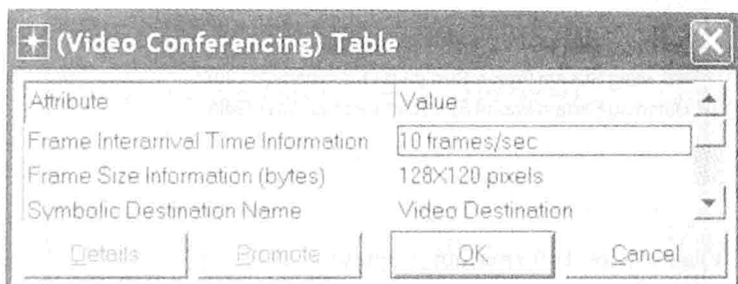


图 5.15　视频会议配置表

1）**Frame Interarrival Time Information**（帧到达间隔时间信息）是一个复合属性，指定进入和外发流量的视频帧到达的频率。如果选择预定义值之一，那么帧到达速率在两个方向就是相同的。展开这个属性，就可独立地为每个方向配置帧到达速率。如图 5.16 所示，复合属性 **Frame Interarrival Time Information** 的定义由两个子属性组成，以 s 为单位指定视频帧进入和外发连续流的到达间隔时间。属性 **Incoming Stream Interarrival Time**（**seconds**）［进入连续流到达间隔时间（秒）］定义在进入连续流中两个连续帧之间的时间，而 **Outgoing Stream Interarrival Time**（**seconds**）［外发连续流到达间隔时间（秒）］指定外发连续流中两个连续视频帧之间的时间。产生下一帧的时间是：将“到达间隔”时间加到前一视频帧被产生并从视频应用发送到低层协议的时间。帧到达频率或速率反比于帧到达间隔时间（例如每秒 10 帧的到达频率对应于帧到达间隔时间为 0.1s 的情况，意味着一个帧是每 0.1s 产生的）。

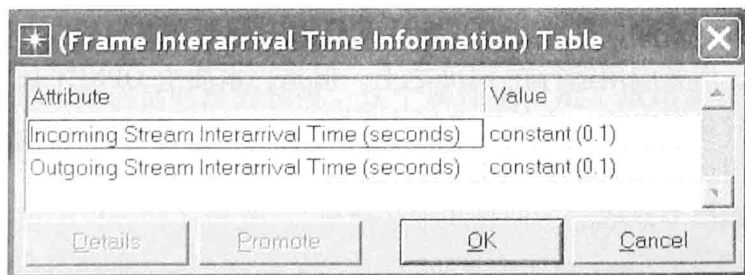


图 5.16　**Frame Interarrival Time Information**（帧到达间隔时间信息）复合属性

2）**Frame Size Information**（**bytes**）［帧尺寸信息（字节）］也是一个复合属性。以字节为单位指定每个进入（即在目的节点产生的）和外发（即在源节点产生的）帧的尺寸。这个属性的预定义值是以像素为单位指定的（例如，在图 5.15 中，帧尺寸为 128 × 120 个像素）。但是，如图 5.17 所示，这个复合属性的子属性是以字节（B）为单位指定帧尺寸的。假定是这样的，即每个像素要求 9 比特（bit），因此一个 128 × 120 个像素的帧要求 128 × 120 × 9bit = 138 240bit，这对应 17 280B。如图 5.17 所示，复合属性

Frame Size Information（**bytes**）由两个子属性组成：**Incoming Stream Frame Size**（**bytes**）［进入连续流帧尺寸（字节）］和 **Outgoing Stream Frame Size**（**bytes**）［外发连续流帧尺寸（字节）］，分别指定进入连续流和外发连续流的视频帧的尺寸。

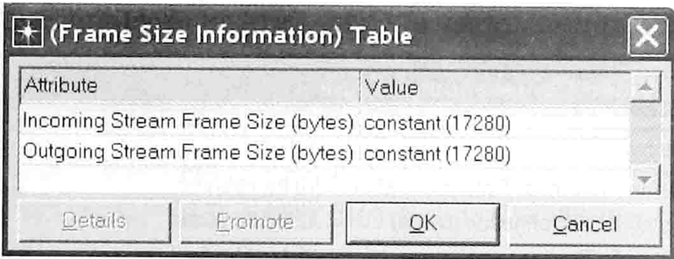


图 5. 17　**Frame Size Information**（**bytes**）［帧尺寸信息（字节）］复合属性

5. 4. 8　话音

OPNET 的话音应用对使用一个数字化话音信号的两个客户端之间的网络通信进行建模。这个应用的定义是十分复杂的，在 OPNET 中存在的针对话音应用的所有配置特征的完整描述会多达数十页。本节仅提供了为建模一项话音应用，在 OPNET 中存在的主要特征的一个概述[⊖]。典型情况下，一个话音呼叫由讲话突发后跟多段静音组成。在话音应用的定义中可显式配置静音和讲话突发的长度。

典型情况下，从一个呼叫者到另一个呼叫者的话音信号传输花费的总时间（即口到耳延迟）由如下各段时间组成：在一端对话音数据的编码和报文化处理，压缩话音报文，通过网络传输编码过的和压缩过的话音数据报文，解压缩接收到的话音数据报文，并最终在另一端处将之解码用于回放。编码和解码延迟是由为这个话音呼叫指定的编码方案的类型确定的。为配置话音应用，OPNET 提供了大量的编码方案模型。压缩和解压缩延迟也可通过话音应用定义的属性进行显式配置。通过网络传输报文化的话音数据的时间，取决于实际的网络配置，在话音应用中是不会指定的。

默认情况下，话音应用运行在 UDP 之上。但是，本质上 OPNET 仿真是使用实时协议（RTP）传输话音报文的，这不要求额外配置。

通常话音应用运行在两个客户端节点之间，不要求存在一个服务器节点。OPNET 支持将话音流量建模为离散的（即显式报文交换）、背景（即分析建模）或这两者的组合。图 5. 18 给出 *Voice Configuration Table*（话音配置表），下面是对每个属性的一个简短解释。

1）**Silence Length**（**seconds**）［静音长度（秒）］是一个复合属性，定义了进出流量连续流的话音会话过程中静音时段的长度。如图 5. 19 所示，默认情况下，对于进出流量，静音时段的长度是使用均值输出为 0. 65s 的指数分布计算得到的。如果需要的话，

⊖ 下面的文章包含有关在 OPNET 中部署 VoIP 应用的更多信息：Salah，K. 和 Alkhoraidly，A.，An OPNET－based simulation approach for deploying VoIP（一种基于 OPNET 的部署 VoIP 的仿真方法），International Journal of Network management，16（3）（2006 年 5 月），159－183。

可改变这些值。静音的进出时段（即在呼叫的任何一方的静音）是相互独立的，这意味着两个客户端同时讲话或同时静音，或呼叫的一侧讲话而另一侧静音都是可能的。

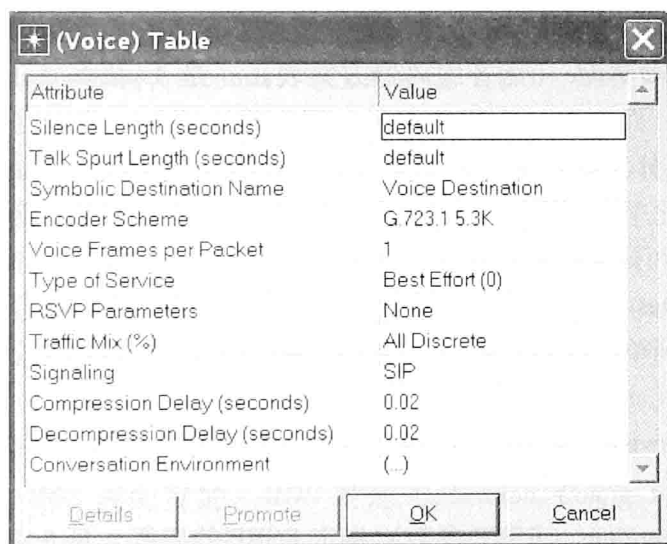


图 5.18　话音配置表

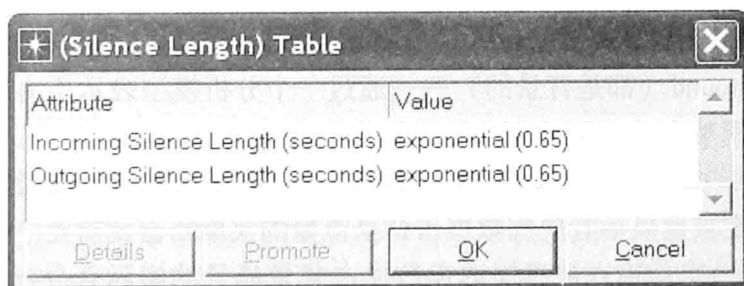


图 5.19　**Silence Length（seconds）**［静音长度（秒）］复合属性

2）**Talk Spurt Length（seconds）**［讲话突发长度（秒）］是一个复合属性，定义话音会话过程中不中断讲话时段的长度。这个属性也指定了进出流量的讲话突发的长度。如图 5.20 所示，默认情况下，对于进出流量，讲话突发的长度是使用均值输出为 0.352s 的指数分布计算得到的。如果需要，这些值也可改变。

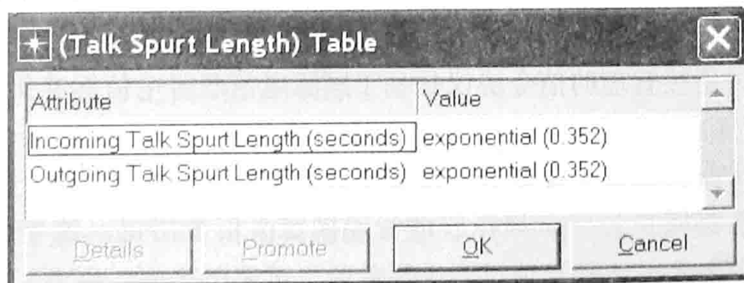


图 5.20　**Talk Spurt Length（seconds）**［讲话突发长度（秒）］复合属性

3）**Symbolic Destination Name**（符号目的地名）指定参与客户端的这次话音会话

的一个目的地节点的符号名字（见 5.4.1 节）。

4）**Encoder Scheme**（编码器方案）指定将话音编码为一个数字信号所用的算法。OPNET 提供了各种编码方案的 30 个预配置模型。可使用一个现有编码方案或通过修改 *Voice Encoder Schemes Table*（话音编码器方案表）［是 *Application Config*（应用配置）对象中的属性之一］而创建一个新的编码方案。话音编码方案是通过属性定义的，如编解码类型（如 PCM、ADPCM、QCELP、VSELP 等）、帧尺寸、预查（lookahead）尺寸、数字信号处理比率、编码速率和讲话活动监测。话音编码是一个独立的话题，由于篇幅限制，在本书中不做讨论。

5）**Voice Frames per Packet**（每个报文的话音帧数）指定话音数据被发送到低层协议之前，被放置到单个应用层报文中的编码话音帧数。低层协议会且经常将应用层报文分片为较小的块（chunks）。

6）**Type of Service**（服务类型）指定在 IP 报文首部中 ToS/DiffServ 字段的值（见 5.4.1 节）。

7）**RSVP Parameters**（RSVP 参数）指定 RSVP 的配置（见 5.4.1 节）。

8）**Traffic Mix**（混合流量）(%) 定义在仿真过程中如何对话音流量建模。这个属性包含如下预设值：

① All Discrete（都是离散的）——所有话音数据都仅被建模为显式流量。

② All Background（都是背景的）——通过一个分析模型表示所有话音数据，其中没有显式话音数据被发送到网络中。

③ 25%、50% 或 75%——表明话音流量使用离散流量和背景流量的混合流量进行建模。属性值指定被建模话音应用数据占背景流量的实际百分比。

④ Edit...——支持输入被建模话音数据占背景流量的实际百分比值，其中 0% 指定话音数据将被表示为所有的离散流量，而 100% 表示话音数据将被建模为所有都是背景流量。

9）**Signaling**（信令）是一个复合属性，指定话音连接如何连接和终止。这个属性包含如下预设值：

① None（无）——不存在管理话音连接的协议。

② SIP——这个话音呼叫的连接将使用 RFC 2543 中定义的会话初始协议（SIP）建立和释放。

③ H.323——连接管理的信令协议是基于国际电信联盟电信标准化部门（ITU－T）的建议之上的。

④ Edit···——支持为这个复合属性设置子属性值：

- **Protocol**（协议）为建立和释放话音连接定义信令协议。这个属性的可能值是 None、SIP 或 H.323。

- **Traffic Modeling**（流量建模）指定将对话音流量的哪部分进行建模。这个子属性有两个可能的值：Control Plane Only（仅控制平面）意味着仅对与信令协议相关联的控制报文进行建模。如果被选择的话，那么在连接处于活跃状态时，将不对应用流量进

行建模（从连接被建立的时间直到连接被释放的时间）。而 Control and Traffic Plane（控制和流量平面）意味着将对信令协议的控制报文和话音流量的数据报文进行建模。这是复合属性 **Signaling**（信令）所有预设值的默认设置。

10）**Compression Delay**（**seconds**）［压缩延迟（秒）］属性指定压缩一个话音报文所花费的时间。

11）**Decompression Delay**（**seconds**）［解压缩延迟（秒）］属性指定解压缩一个话音报文所花费的时间。

12）**Conversation Environment**（会话环境）指定在话音呼叫的两端配置环境的质量。会话环境的质量可被设置为如下预设值之一：Land phone – Quiet room（陆地电话 – 安静的房间）、Land phone – Noisy room（陆地电话 – 吵闹的房间）、Cell phone in building（建筑内的蜂窝电话）、Cell phone in SUV or sedan（SUV 或小轿车中的蜂窝电话）和 Cell phone in convertible（折篷汽车中的蜂窝电话）。

这些值是通过 *MOS* 表配置的，*MOS* 表是 *Application Config* 对象中的属性之一。MOS 表示 **Mean Opinion Score**（平均意见得分），是评估一个话音呼叫质量的一项度量措施。

5.5 使用符号节点名字

在 OPNET 中，当定义标准应用时，应用的服务器和目的地是通过符号名字而不是在被仿真网络中实际节点的名字来引用的。因此，可选择诸如“FTP Server”的一个符号名字，当在被仿真网络中部署应用时，将符号名字映射到一个或多个具体的节点。同样的思路用于指定定制应用，其中并不尝试预测哪些节点将作为定制应用流量的源和目的地而运行，仅需要通过符号名字［如“Custom HTTP Login Source”（定制的 HTTP 登录源）和“IM Authentication Server”（IM 认证服务器）］指定源和目的地。这个特征将应用定义与实际网络拓扑的配置分离开来，并得到配置仿真场景的一种灵活的架构。考虑如下情况：一名网络架构师创建了在各种不同的网络拓扑之上由几个应用组成的一项仿真研究。使用“symbolic name”（符号名字）特征，OPNET 能够独立于实际的网络拓扑而维护应用源和目的地的名字。结果是，一旦定义并配置应用，则架构师就能够以任何期望的网络拓扑直接部署所定义的应用，方法是指定符号客户端和/或服务器名字与新拓扑中实际节点的名字之间的映射关系。

在 OPNET 中，仅有高级节点模型（即节点模型的名字包括扩展名_ *adv*）支持实际名字和符号名字之间的一个映射，方法是显式地改变属性值。典型情况下，当部署用户概要时，这个步骤是由 OPNET 自动实施的。进而，OPNET 也在源符号名字和目的地符号名字之间做出了区分。典型情况下，源符号名字仅由定制应用来指定一个名字，它识别作为一个定制应用阶段的源发者或发起者的节点。目的地符号名字被用来指定这样的节点，它们作为标准和定制应用的一个服务器或流量目的地。5.5.1 节和 5.5.2 节中的指令提供了将源和目的地符号名字分别映射到实际节点名字的步骤。这些指令仅适用于

端节点模型，如工作站、服务器、LAN等。OPNET通过称为 **Application: Source Preferences**（应用：源首选项）和 **Application: Destination Preferences**（应用：目的地首选项）的节点属性，指定实际节点名字和相应源或目的地符号名字之间的映射关系，如图5.21所示。

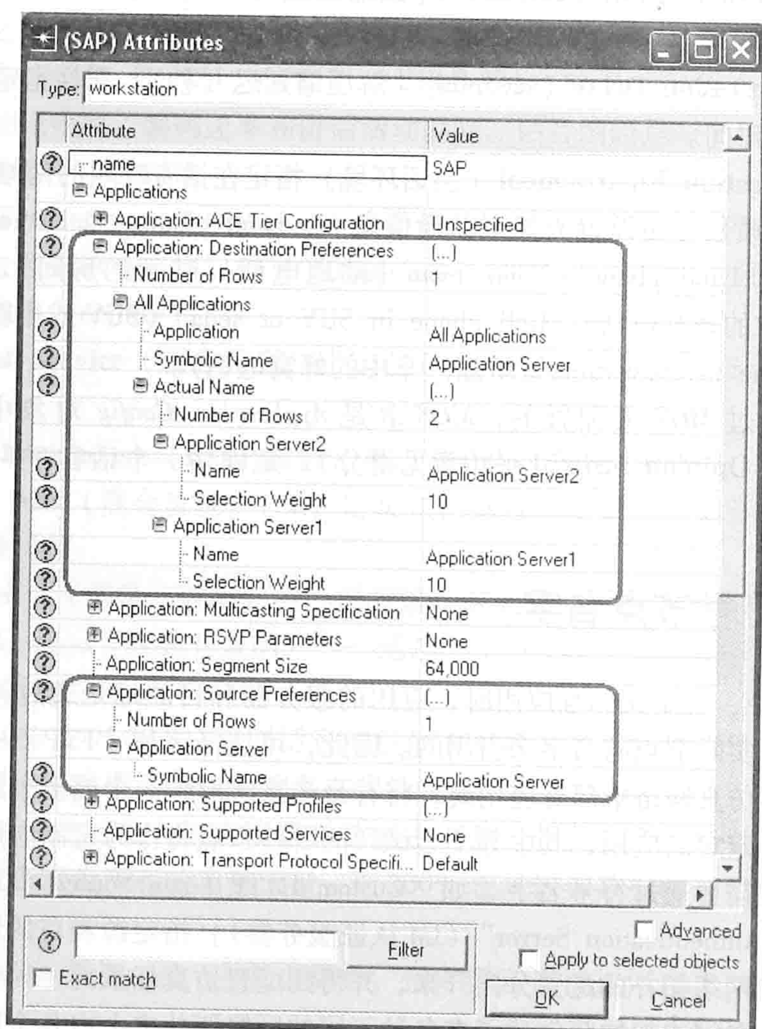


图5.21 **Application: Source Preferences**（应用：源首选项）和 **Application: Destination Preferences**（应用：目的地首选项）属性

5.5.1 手动配置一个应用的源首选项

这个属性仅适用于定制应用。它指定当前节点将作为一个定制应用任务中一个阶段的发起者或启动者，其属性 **Task Specification... Manual Configuration... Source**（任务规格... 手动配置... 源）匹配所指定的符号名字。欲了解定义定制应用、其任务和阶段的更多细节，参见6.2节。遵循这些步骤，将符号源名字映射到实际的节点名字：

1）在被仿真系统的网络拓扑中，在将作为相应应用的一个源的节点上右击，并从出现的菜单中选择 **Edit Attributes**（编辑属性）选项。

2）展开属性 **Application... Application: Source Preferences**（应用...应用：源首选项）。注意这个属性仅存在于高级节点模型中（即以扩展名_ adv 作为其名字组成部分的节点模型）。

3）设置属性 **Number of Rows**（行数）的值，这对应于映射到这个节点的符号名的数量。单个节点作为多项应用的源运行，这是可能的。

4）对于出现的每个新行，展开复合属性 **None**（无），并将属性 **Symbolic Name**（符号名字）的值改变为可用符号名字下拉列表中的值之一。这种改变将使当前节点作为满足如下条件的所有应用的一个源，这些应用的属性 **Source** 设置为相应的符号名字。如果多个节点被映射到同一符号源名，那么就随机地选择实际节点。

5.5.2 手动配置一个应用的目的地首选项

这个属性适用于定制应用和标准应用。它提供了一个符号目的地名字和网络中实际节点之间的一个映射关系。如果多个节点匹配单个目的地符号名字，那么基于关联的选择权重，选择目的地节点。如果没有指定目的地符号名字和实际节点之间的一个映射关系，那么将从网络中支持所指定应用之服务的所有节点中，随机选择目的地节点。遵循这些步骤，将符号目的地名字映射到实际节点名字：

1）在被仿真系统的网络拓扑中，右击将作为一个源的节点，从出现的菜单中选择 **Edit Attributes** 选项。注意在源节点处源和目的地首选项都要配置。

2）展开 **Applications... Application: Destination Preferences**（应用...应用：目的地首选项）属性。这个属性仅存在于高级节点模型中（即以扩展名_ *adv* 作为其名字组成部分的节点模型）。

3）设置属性 **Number of Rows** 的值，对应于希望在符号服务器名和实际节点名之间提供一个映射的应用数量。在每个应用的基础上，实施服务器或目的地映射操作。

4）为每个新出现的行（是一个复合属性），实施如下动作。新行都将有默认名字 All Applications（所有应用），指明符号服务器名字和仿真中的实际物理节点之间的映射规则都将适用于所有定义的应用。

① 展开行。

② 设置属性 **Application** 的值，默认情况下设置为 All Applications。这个属性指定这个映射规则将应用到的应用的名字。这个属性有助于在以同一符号服务器名字定义的各应用间做出区分。这个属性的值字段包含当前在 *Application Config*（应用配置）工具对象中定义的所有应用的一个下拉列表。可仅将属性 **Application** 设置为那个列表中的值之一。

③ 设置属性 *Symbolic Name*（符号名字）的值，这指定要被映射到一个实际节点名字的符号名字。同样，下拉列表提供了 *Task Config*（任务配置）和 *Application Config*（应用配置）工具对象中定义的所有符号名字。

④ 配置属性 **Actual Name**（实际名字），指定映射到这个符号服务器名字、实际节点名和选择权重（服务器节点将被选择的概率）的实际节点的数量。例如，如图 5.21

所示，目的地的符号名 Application Server 被映射到被仿真网络中的两个实际节点，称为 Application Server2 和 Application Server1。具体而言，这个复合属性包含如下子属性：

- **Number of Rows**（行数）指定可作为这项应用的一个服务器运行的节点总数。为每台服务器将创建一个新行。
- 对于每行，指定到一个实际服务器的映射，方法是设置如下两个属性的值：**Name**（名字）是在被仿真系统中实际节点的名字，节点将作为一台服务器运行。这个属性提供在系统中存在的所有节点的名字的一个列表。必须仅选择支持当前应用的那些节点，否则，这个配置设置将被忽略。而 **Selection Weight**（选择权重）是服务器选择权重，确定在应用执行过程中这台服务器将被联系的概率。

5.6 应用统计

OPNET 中的每个标准应用都有一组统计量，可用于仿真过程中的数据收集。典型情况下，这些统计量描述应用的性能，是以各种应用延迟性质以及流量到达和离开率度量的。例如，电子邮件应用包括如下统计量：下载和上传响应时间，以及以字节每秒和报文数每秒为单位的流量接收和发送率。

全局应用统计量描述网络中整体的应用性能。例如，电子邮件应用的 **Traffic Received（bytes/sec）**［接收到的流量（字节/秒）］全局统计量，收集来自传输层的数据，它以在运行电子邮件应用的所有节点上（即客户端和服务器）平均得到的字节数每秒为单位，度量到达率。节点应用统计在客户端节点和服务器节点之间做出区分，并收集一个特定节点处有关应用性能的信息。每个标准应用包含一组统计量，描述在客户端节点和服务器节点处记录的应用性能。例如，OPNET 用下面两类收集电子邮件应用的统计量：

1）**Client E-mail**（客户端电子邮件），在客户端节点处收集有关应用性能的统计量，例如，发送一条请求和接收一条响应之间经历的延迟。

2）**Server E-mail**（服务器电子邮件），在服务器节点处收集有关电子邮件应用性能的统计量，例如，在服务器处电子邮件请求到达率或服务器处理一条应用请求所花费的时间。

如果仿真模型不包含一个 *Application Config* 节点或需求对象，则客户端统计是不可用于收集的。如果在仿真模型中没有服务器节点，则服务器统计量是不可用于收集的。

可用应用统计量的汇总在表 5.3 和表 5.4 中给出。为了避免重复，将几个常用统计类组合到单一小节。在这样的情况下，除非另外指出，否则统计描述适用于那一小节中的所有统计类。对每个这样的类，存在一个独立的统计量，且它仅为单独属于那个类的流量计算统计值。例如，对 **DB Entry** 和 **DB Query** 类，存在全局统计量 **Traffic Received（bytes/sec）**。这个统计量测量部署于网络中的所有数据库应用接收到的平均流量速率（以字节数每秒为单位）。但是，为数据库输入和数据库查询操作单独计算统计量。

表5.3 全局应用统计汇总

类别	名字	描述
DB Entry，DB Query（DB输入，DB查询）	*Response Time* (*sec*)［响应时间（秒）］	作为对一条请求报文的应答，一条数据库响应报文到达所花费的时间
	Traffic Received (*bytes/sec*)［接收到的流量（字节/秒）］ *Traffic Received* (*packets/sec*)［接收到的流量（报文数/秒）］	到部署于网络中的所有数据库应用处的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的
	Traffic Sent (*bytes/sec*)［发送的流量（字节/秒）］ *Traffic Sent* (*packets/sec*)［发送的流量（报文数/秒）］	从部署于网络中的所有数据库应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的
Email（电子邮件）	*Download Response Time* (*sec*)［下载响应时间（秒）］	作为对一条客户端请求的响应，从服务器到达的电子邮件消息花费的时间。该时间包括连接建立延迟
	Traffic Received (*bytes/sec*)［接收到的流量（字节/秒）］ *Traffic Received* (*packets/sec*)［接收到的流量（报文数/秒）］	到部署于网络中的所有电子邮件应用处的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的
	Traffic Sent (*bytes/sec*)［发送的流量（字节/秒）］ *Traffic Sent* (*packets/sec*)［发送的流量（报文数/秒）］	从部署于网络中的所有电子邮件应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的
	Upload Response Time (*sec*)［上传响应时间（秒）］	作为对客户端上传电子邮件消息的响应，从电子邮件服务器到达的一条ACK所花费的时间。该时间包括连接建立延迟
FTP	*Download Response Time* (*sec*)［下载响应时间（秒）］	从服务器下载一个文件所花费的时间，是从客户端发出一条get请求到文件下载完成的时间。这个时间包括连接建立和拆除延迟。仅当连接被关闭时，统计量才被更新
	Traffic Received (*bytes/sec*)［接收到的流量（字节/秒）］ *Traffic Received* (*packets/sec*)［接收到的流量（报文数/秒）］	到部署于网络中的所有FTP应用处的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的

（续）

类别	名字	描述
FTP	*Traffic Sent*（*bytes/sec*）［发送的流量（字节/秒）］ *Traffic Sent*（*packets/sec*）［发送的流量（报文数/秒）］	从部署于网络中的所有 FTP 应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的
	Upload Response Time（*sec*）［上传响应时间（秒）］	将一个文件上传到服务器所花费的时间，从客户端开始上传文件（即发出 put 请求）到一个 ACK 从服务器到达的时间。该时间包括连接建立延迟和拆除延迟。该统计量仅在连接被关闭后才更新
HTTP	*Object Response Time*（*sec*）［对象响应时间（秒）］	从 HTTP 页面检索单个内联对象所花费的时间
	Page Response Time（*sec*）［页面响应时间（秒）］	检索包括所有内联对象的一个完整 HTML 页面所花费的时间
	Traffic Received（*bytes/sec*）［接收到的流量（字节/秒）］ *Traffic Received*（*packets/sec*）［接收到的流量（报文数/秒）］	到部署于网络中的所有 HTTP 应用处的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的
	Traffic Sent（*bytes/sec*）［发送的流量（字节/秒）］ *Traffic Sent*（*packets/sec*）［发送的流量（报文数/秒）］	从部署于网络中的所有 HTTP 应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的
Print	*Traffic Received*（*bytes/sec*）［接收到的流量（字节/秒）］ *Traffic Received*（*packets/sec*）［接收到的流量（报文数/秒）］	到部署于网络中的所有 Print（打印）应用处的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的
	Traffic Sent（*bytes/sec*）［发送的流量（字节/秒）］ *Traffic Sent*（*packets/sec*）［发送的流量（报文数/秒）］	从部署于网络中的所有 Print（打印）应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的

（续）

类别	名字	描述
Remote Login（远端登录）	*Response Time*（*sec*）［响应时间（秒）］	作为对一条请求报文的应答，到达一条远程登录响应报文所花费的时间
	Traffic Received（*bytes/sec*）［接收到的流量（字节/秒）］ *Traffic Received*（*packets/sec*）［接收到的流量（报文数/秒）］	到部署于网络中的所有远程登录应用处的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的
	Traffic Sent（*bytes/sec*）［发送的流量（字节/秒）］ *Traffic Sent*（*packets/sec*）［发送的流量（报文数/秒）］	从部署于网络中的所有远程登录应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的
Video Conferencing（视频会议）	*Packet Delay Variation*（报文延迟变化）	从视频报文被产生直到被接收的时间期间，视频报文所经历的端到端时延变化
	Packet End - to - End Delay（报文端到端时延）	从报文在源处被创建至到达目的地节点的视频报文所花费的时间
	Traffic Received（*bytes/sec*）［接收到的流量（字节/秒）］ *Traffic Received*（*packets/sec*）［接收到的流量（报文数/秒）］	到部署于网络中的所有视频会议应用处的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的
	Traffic Sent（*bytes/sec*）［发送的流量（字节/秒）］ *Traffic Sent*（*packets/sec*）［发送的流量（报文数/秒）］	从部署于网络中的所有视频会议应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的
Voice（话音）	*Jitter*（*sec*）［抖动（秒）］	如果 $Tc(i)$ 是报文 i 和 $i+1$ 处在源节点被创建的时间差，$Tp(i)$ 为报文 i 和 $i+1$ 在目的地节点被回放的时间差，那么 $Jitter = Tp(i) - Tc(i)$
	MOS Value（MOS 值）	记录的话音流量平均意见得分（MOS）

(续)

类别	名字	描述
Voice(话音)	*Packet Delay Variation*(报文延迟变化)	从话音报文被创建直到它们被接收,话音报文所经历的端到端延迟变化
	Packet End - to - End Delay(*sec*)[报文端到端延迟(秒)]	话音报文所经历的总延迟[即analog - to - analog(模拟到模拟)或mouth - to - ear(口到耳)延迟]。它包括网络、编码/解码和(解)压缩延迟。这个统计量记录从网络中所有节点处收集的数据
	Traffic Received(*bytes/sec*)[接收到的流量(字节/秒)] *Traffic Received*(*packets/sec*)[接收到的流量(报文数/秒)]	到部署于网络中的所有话音应用处的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的
	Traffic Sent(*bytes/sec*)[发送的流量(字节/秒)] *Traffic Sent*(*packets/sec*)[发送的流量(报文数/秒)]	从部署于网络中的所有话音应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的

表5.4 节点应用统计汇总

类别	名字	描述
Client DB, Client DB Entry, Client DB Query(客户端DB、客户端DB输入,客户端DB查询)	*Response Time*(*sec*)[响应时间(秒)]	对这个节点中的所有数据库应用,作为对一条DB输入/查询请求报文的响应,一条数据库响应报文到达所花费的时间。这个统计量不可用于客户端DB类
	Traffic Received(*bytes/sec*)[接收到的流量(字节/秒)] *Traffic Received*(*packets/sec*)[接收到的流量(报文数/秒)]	到部署于该节点处的所有数据库应用的数据库(输入和查询)/输入/查询请求的平均流量到达率。该统计量是基于从传输层到达的应用数据计算得到的
	Traffic Sent(*bytes/sec*)[发送的流量(字节/秒)] *Traffic Sent*(*packets/sec*)[发送的流量(报文数/秒)]	从部署于该节点处的所有数据库应用的数据库(输入和查询)/输入/查询请求的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的
	Transaction Size(*bytes*)[事务尺寸(字节)]	对于这个节点中所有数据库应用,由数据库输入/查询请求产生的事务报文的尺寸

（续）

类别	名字	描述
Server DB, Server DB Entry, Server DB Query（服务器DB、服务器DB输入，服务器DB查询）	*Load*（*requests/sec*）[负载（请求/秒）]	在服务器节点处数据库（输入和请求）/输入/查询请求的到达率。这些请求可属于不同数据库会话（可源于网络中的不同节点）
	Load（*session/sec*）[负载（会话/秒）]	在这台数据库服务器上当前活跃的会话数。这个统计量仅适用于Server DB类
	Task Processing Time（*sec*）[任务处理时间（秒）]	这个服务器节点处理一个数据库（输入和查询）/输入/查询请求所花费的时间。这个统计量是这样计算的，从服务器完成这条查询的时间减去请求到达的时间
	Traffic Received（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）] *Traffic Sent*（*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	这些服务器DB统计量的每项与相应的客户端DB统计量具有相同含义，例外情况是，它们是在支持数据库应用的单个特定服务器节点上收集的
Client Email（电子邮件）	*Download Response Time*（*sec*）[下载响应时间（秒）] *Traffic Received*（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）] *Traffic Sent*（*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）] *Upload Response Time*（*sec*）[上传响应时间（秒）]	客户端电子邮件统计量中的每项都与相应的全局电子邮件统计量的含义相同，例外情况是，这些统计量仅是在运行电子邮件应用服务的单个特定客户端节点上收集的

（续）

类别	名字	描述
Client FTP	*Download Response Time*（*sec*）[下载响应时间（秒）] *Traffic Received*（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）] *Traffic Sent*（*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）] *Upload Response Time*（*sec*）[上传响应时间（秒）]	客户端FTP统计量中的每项都与相应的全局FTP统计量的含义相同，例外情况是，这些统计量仅是在运行FTP应用服务的单个特定客户端节点上收集的
	Download File Size（*bytes*）[下载文件尺寸（字节）] *Upload File Size*（*bytes*）[上传文件尺寸（字节）]	由客户端节点接收/发送的下载/上传文件尺寸
Client HTTP	*Object Response Time*（*sec*）[对象响应时间（秒）] *Page Response Time*（*sec*）[页面响应时间）（秒）] *Traffic Received*（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）] *Traffic Sent*（*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	客户端HTTP统计量中的每项都与相应的全局HTTP统计量的含义相同，例外情况是，这些统计量仅是在运行HTTP应用服务的单个特定客户端节点上收集的
	Downloaded Objects，*Downloaded Pages*（下载的对象，下载的页面）	下载的个体内联对象或完整的HTML页面总数
	User Cancelled Connections（用户取消的连接数）	这个统计量收集页面下载被取消的次数，原因是用户开始了一个新的页面下载而以前的页面仍然在下载

（续）

类别	名字	描述
Client Print	*File Size* (*bytes*) [文件尺寸（字节）]	发送到打印机的任务平均尺寸
	Traffic Sent (*bytes/sec*) [发送的流量（字节/秒）] *Traffic Sent* (*packets/sec*) [发送的流量（报文数/秒）]	从部署于这个节点中的 Print（打印）应用处的平均流量离开率。该统计量是基于转发到传输层的应用数据计算得到的
Client Remote Login（客户端远端登录）	*Response Time* (*sec*) [响应时间（秒）] *Traffic Received* (*bytes/sec*) [接收到的流量（字节/秒）] *Traffic Received* (*packets/sec*) [接收到的流量（报文数/秒）] *Traffic Sent* (*bytes/sec*) [发送的流量（字节/秒）] *Traffic Sent* (*packets/sec*) [发送的流量（报文数/秒）]	这些客户端远程登录统计量中的每项都与相应的全局远程登录统计项具有相同含义，例外情况是，这些统计量是仅在运行远端登录应用的单个特定客户端节点上收集的
Server E－mail，Server FTP，Server HTTP，Server Remote Login，Server Print	*Load* (*requests/sec*) [负载（请求数/秒）]	在服务器节点处客户端请求的到达率。这些请求可属于源自网络中不同节点的不同会话
	Load (*sessions/sec*) [负载（会话数/秒）]	收集在这台服务器处当前活跃的会话数
	Task Processing Time (*sec*) [任务处理时间（秒）]	这个服务器节点处理一条客户端请求花费的时间。这个统计量是这样计算的，从服务器完成处理这条请求的时间减去请求到达的时间
	Traffic Received (*bytes/sec*) [接收到的流量（字节/秒）] *Traffic Received* (*packets/sec*) [接收到的流量（报文数/秒）] *Traffic Sent* (*bytes/sec*) [发送的流量（字节/秒）] *Traffic Sent* (*packets/sec*) [发送的流量（报文数/秒）]	这些服务器统计量中的每项都与相应客户端统计量的含义相同，例外情况是，它们是在支持相应应用服务的单个特定服务器节点上收集的。发送的流量这个统计量不可用于服务器打印（Server Print）统计类

（续）

类别	名字	描述
Video Called Party（视频被叫方），**Video Calling Party**（视频呼叫方），**Video Conferencing**（视频会议）	*Packet Delay Variation*（报文延迟变化） *Packet End-to-End Delay*（*sec*）[报文端到端时延（秒）] *Traffic Received*（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）] *Traffic Sent*（*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	这些统计量中的每项都与相应全局视频会议的统计量含义相同，例外情况是，它们是在支持视频会议应用的单个特定节点上收集的。这些统计类之间的区别如下： • Video Called Party——在每个呼叫者基础上为被叫方收集统计信息 • Video Calling Party——在每个呼叫者基础上为呼叫方收集统计信息 • Video Conferencing——为这个节点上的所有视频流量收集统计信息
Voice Application（话音应用），**Voice Called Party**（话音被叫方），**Voice Calling Party**（话音呼叫方）	*MOS Dejitter Delay*（*sec*）[MOS去抖动延迟（秒）]	当报文在网络上发送时由去抖动导致的以秒为单位的延迟
	MOS Dejitter Loss Rate（MOS去抖动丢失率）	报文丢失比率计算为报文丢失数量与报文总数的比率，其中丢失是由于到达间隔延迟大于去抖动延迟造成的
	MOS Network Loss Rate（MOS网络丢失率）	报文丢失率计算为由于某些网络条件造成的报文丢失数量与发送的报文总数的比率
	MOS Value（MOS值）	平均意见得分值
	Packet Delay Variation（报文延迟变化） *Packet End-to-End Delay*（*sec*）[报文端到端延迟（秒）] *Traffic Received*（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）] *Traffic Sent*（*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	这些统计量中的每项都与相应全局话音的统计量含义相同，例外情况是，它们是在支持话音应用的单个特定节点上收集的。这些统计类之间的区别如下： • Video Application——在这个节点上为所有话音流量收集统计信息 • Voice Called Party——在每个呼叫者基础上为被叫方收集统计信息 • Voice Calling Party——在每个呼叫者基础上为呼叫方收集统计信息

第6章　高级流量生成功能特征

6.1　定制应用简介

标准应用模型是非常容易配置的，且它们为常用的应用（如电子邮件、FTP和远程登录）的建模提供足够的细节。但是，标准应用仅对一个**two - tier**（两层）通信范型进行建模，不支持对被仿真应用协议的修改。为了解决这个问题，OPNET也为建模**custom applications**（定制应用）提供了设施，可表示遵循一个用户定义协议的非标准、多层应用。例如，电子邮件的标准应用模型仿真一个两层客户端—服务器消息交换，其中客户端周期性地将新撰写的消息发送到它的电子邮件服务器，也独立地轮询服务器，检索目标为本客户端的电子邮件。但是，在实际生活中，一个电子邮件应用的行为是稍稍有些不同的。从一个客户端到另一个客户端发送一条电子邮件消息主要包括如下步骤：

1）客户端A撰写发送客户端B的一条电子邮件消息。

2）客户端A将这条电子邮件消息上传到其电子邮件服务器。

3）客户端A的电子邮件服务器中继该消息到客户端B的电子邮件服务器。

4）客户端B从其电子邮件服务器下载由客户端A发送的这条电子邮件消息。

上述行为代表一个**multitier**（多层）消息交换，其中在客户端A和B之间通过其电子邮件服务器存在一个复杂的交互序列。修改标准的电子邮件应用模型，来仿真这样的一个电子邮件协议，是一项非常具有挑战性的工作，这也许将要求使用OPNET Modeler改变电子邮件应用过程模型的代码。另外，这种行为可容易地使用定制应用建模框架进行仿真，而不需要编写一行代码。

在OPNET中，所有定制应用都是通过一系列**tasks**（任务）加以定义的。每项任务进一步被分为个体**phases**（阶段）。图6.1汇总了定制应用建模框架的架构性层次结构。一个任务的每个阶段代表两个端点之间的请求—响应消息交换或单个端节点上的处理。例如，第一个阶段也许代表一个源发节点和服务器A之间的消息交换，第二个阶段也许对服务器A和服务器B之间的消息交换进行建模，等等，这构造了连续的阶段，其中每个端节点代表一个独立的通信层。另外，典型情况下，作为一个阶段中目的地的端节点可作为后一阶段中消息交换的一个源。但是，发起任务的节点（即在任务内第一阶段的源/源发者）也许未必是任务内最后阶段的目的节点，虽然非常常见的情况是，一个定制应用，使在第一阶段中的源与最后阶段的目的地为同一端节点。

考虑在互联网上购买物品的一项应用的例子，要求如下步骤。

1. Login（登录）

1）客户端将其机密信息发送到Web服务器。

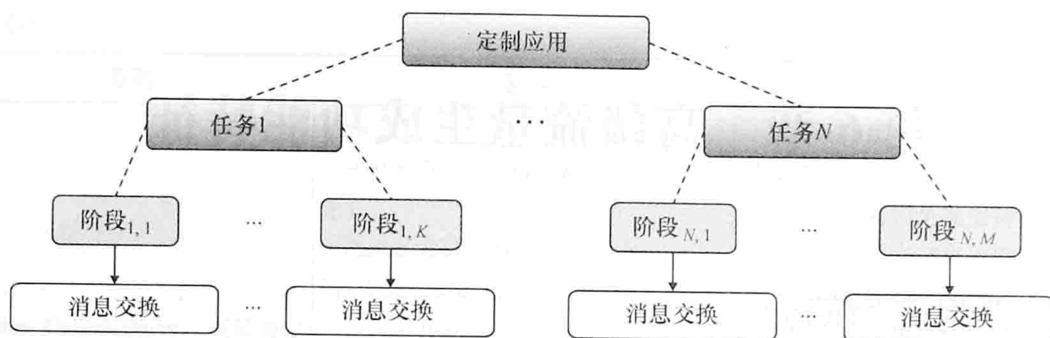


图 6.1　定制应用层次结构

2）Web 服务器联系一台认证服务器验证客户端的机密信息。

3）认证服务器处理提供的机密信息，并为 Web 服务器产生一条响应。

4）Web 服务器将一条 ACK 发回客户端，指明登录是否成功。

2. Browsing（浏览）

1）客户端将对一件物品的请求发送到 Web 服务器。

2）Web 服务器搜索其内部数据库，并将搜索的结果发回到客户端。

3）在客户端浏览物品的清单时，这个过程被重复多次。

3. Purchase（购买）

1）客户端提供给 Web 服务器的是它希望购买的物品列表和它的信用卡信息。

2）Web 服务器联系银行服务器，验证信用卡信息的有效性。

3）Web 服务器联系数据库服务器，更新有关被购物品的存储信息。

4）Web 服务器将购买的收条发回客户端。

为了对上面的例子中所述的应用进行建模，列表的每个顶级标题（如 login、browsing 和 purchase）可代表定制应用定义内的一个任务，而每个次级标题可代表相应任务内的一个阶段。在每个阶段过程中个体端节点之间的消息交换可通过指定配置参数（如消息长度、消息产生率和交换的消息总数）进行建模，这些参数类似于标准应用定义中使用的属性。注意在相应定制应用阶段过程中参与信息传输的每个端节点实际上表示一个独立的通信层。

如您所想到的，指定一个定制应用的过程是非常复杂的。出于这个原因，在使用 OPNET 开始一个定制应用的配置之前，要小心地仔细检查研究并指定要建模的定制应用的所有方面。具体而言，识别并定义应用的如下特点是重要的：

1）在这个应用中涉及的端节点或层的数量和类型。

2）组成被建模应用的个体任务。

3）每个任务的个体阶段。

4）对任务的每个阶段：

① 在阶段中涉及的端节点或层。

② 如果阶段是一次数据传输，则是在端节点之间交换的流量的特点。

③ 如果阶段涉及在服务器处的处理，则是处理典型需要的时间总量。

对于上述的定制应用例，描述已经清晰地识别出过程中涉及的端节点（如客户端、Web 服务器、数据库服务器、认证服务器和银行服务器）。该应用也已经被分为个体任务，且针对每项任务的各节点已经清晰地确定。剩下的唯一工作是定义每个阶段过程中交换的流量特点和在登录任务过程中认证处理的长度。如您想到的，这样一个配置的细节是高度应用相关的，并必须在逐个案例的基础上进行处理。希望本节提供了在 OPNET 中定义定制应用中主要步骤的一个不错的概述，现在可有一个应用如何通过任务和阶段表示的基本理解。

一旦确定应用规格，在 OPNET 中配置一项定制应用的过程就由如下步骤组成：

1）使用 *Task Config* 工具对象，配置定制应用的每个任务。

2）使用 *Application Config* 工具对象，配置定制应用。

3）指定在仿真过程中要收集的统计量。

4）使用 *Profile Config* 工具对象，指定采用被配置定制应用的一个用户概要。

5）在被仿真系统中部署所定义的用户概要。

在本章，主要将关注点放在列表中提到的前三项，同时像在第 7 章讨论的那样指定用户概要和部署应用。6.2 节描述配置个体任务和阶段的 OPNET 具体细节。接着是描述使用特定的任务定义如何配置定制应用（见 6.3 节），再后是配置一个定制应用的详细范例（见 6.4 节）。6.5 节和 6.6 节提供了显式流量生成源和流量需求的一个概述。本章以评估各种高级 OPNET 流量生成机制的可用统计量的描述结尾。

6.2　为定制应用配置任务和阶段

指定定制应用的第一步是将 *Task Config* 工具对象添加到项目工作空间中。

6.2.1　Task Config 工具对象

如图 6.2 所示的 *Task Config* 工具对象，位于和 *Application Config* 对象相同的 **Object Palette Tree**（对象调色板树）的文件夹中（见 5.3.1 节）。将 *Task Config* 对象添加到 **Project Editor** 工作控制中的步骤与任何其他对象的步骤相同：

1）识别并选择 **Object Palette Tree** 中的对象。

2）将对象拖放到工作空间中。

与其他工具对象一样，仅需要一个 *Task Config* 节点来配置当前仿真场景内的所有任务。



图 6.2　*Task Config* 工具对象

如图 6.3 所示，*Task Config* 对象仅由两个属性组成：**name**（名字）（指定当对象出现在工作空间中时的名字）和 **Task Specification**（任务规格）（一个复合属性，支持为定制应用指定个体任务）。与定义同一项目的多个实例的许多复合属性常见的情况一样，*Task Specification* 属性包含称为 **Number of Rows**（行数）的一个子属性，它为这个仿真中针对所有定制应用指定要定义的任务数。与通常情况一样，这个属性接受零或

正整数值。零值代表这样一种情况，此时没有任务可用于定制应用。将这个属性设置为一个正整数值，如 N，将得到配置定制应用任务的 N 个复合子属性，出现在属性 **Number of Rows** 下。

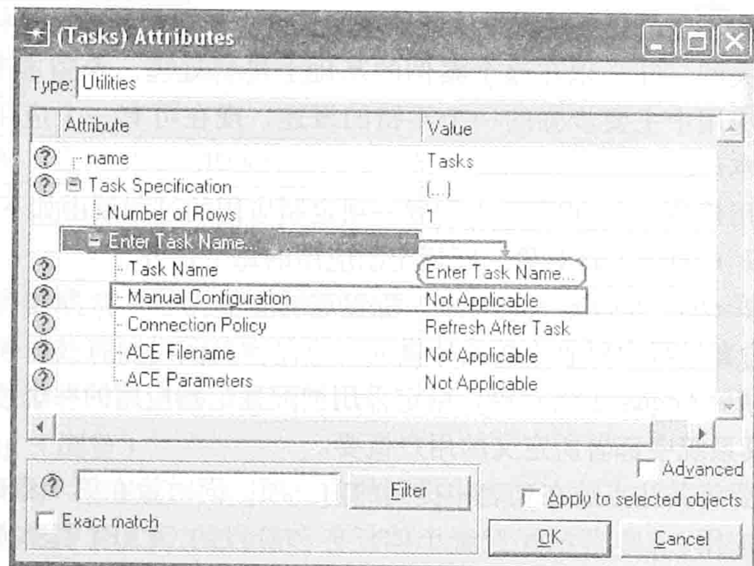


图 6.3　*Task Config* 对象的属性

6.2.2　指定任务定义

Task Config 对象中 **Task Specification** 属性的每行都被用来定义一个单独的任务。当首次创建时，每行（称为一个任务定义）将默认地被命名为 **Enter Task Name...**。如图 6.3 所示，一个任务定义由如下属性组成：

1）**Task Name**（任务名）指定任务的一个字母数字组成的名字。设置这个属性的值，自动地改变包含当前任务定义的整行的名字。

2）**Manual Configuration**（手动配置）通过定义任务的个体阶段过程中要实施的动作，来配置任务行为，这将在后面讨论。

3）**Connection Policy**（连接策略）指定连接重用策略。连接可以是在所有任务间被重用或每个任务都要重置。但是，由这个属性定义的连接重用策略将由为任务的个体阶段定义的连接策略所替换（superseded）。6.2.3 节将详细地讨论这个问题。

4）**ACE Filename**（ACE 文件名）和 **ACE Parameters**（ACE 参数）属性允许使用由应用特征环境（ACE）OPNET 软件程序产生的文件配置定制应用任务。这个话题超出了本书的范围，这里不作讨论。

Manual Configuration 属性表示为一个表的形式，如图 6.4 所示，其中每行指定任务内的单个阶段。*Manual Configuration Table* 中各阶段的数量和顺序是通过称为 *Rows*（行数）的文本框加以控制的。这个文本框接受一个非负整数值，指定要在表内定义的总行数，或阶段数。如下按钮位于文本框的旁边：

1）**Delete**（删除）按钮从表中去除高亮显示的行或阶段。从表中去除一行，也将

由文本框 *Rows* 指定的行数减 1。当行数被设置为 0 时，**Delete** 按钮是不可用的。

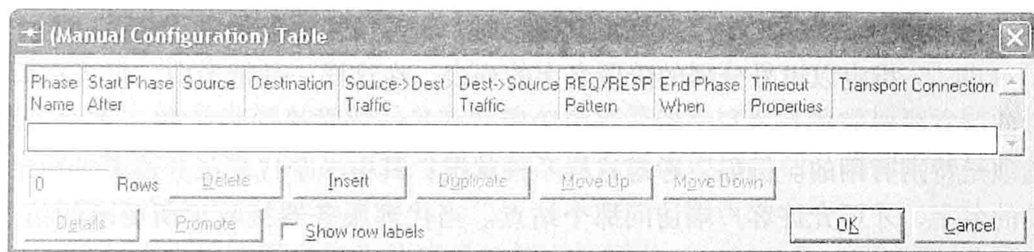


图 6.4　*Manual Configuration Table*（手动配置表）

2）**Insert**（插入）按钮将一行添加到表中。新行被添加到当 **Insert** 按钮被单击时所选择的那行之上。当一个新行被插入到表中时，在文本框 *Rows* 中指定的行数就加 1。

3）**Duplicate**（复制）按钮复制被选中的行。复制的内容被插入到所选中行的下面。当一行被复制时，在文本框 *Rows* 中指定的行数也做加 1 处理。当行数被设置为 0 时，**Duplicate** 按钮是不可用的。

4）**Move Up**（上移）和 **Move Down**（下移）按钮支持通过将行上移或下移而对行的顺序进行操作。当行数被设置为 0 时，这两个按钮都是不可用的（即没有可被上移或下移的行）。

5）**OK** 按钮保存改变，并将包含 *Manual Configuration Table* 的窗口关闭，而 **Cancel** 按钮则在不保存改变的情况下关闭窗口。

6）**Details**（细节）按钮提供所选择列的描述。

7）**Promote**（提升）按钮将所选单元的值提升到下一个更高层次。

6.2.3　指定阶段配置

Manual Configuration Table 的列代表各阶段的配置属性，同时表中的每行对应于单个阶段的一个完全配置。现在描述每列的含义。

第 1 列 **Phase Name**（阶段名）指定任务和统计组（阶段所属的（可选））内一个阶段的字母数字组成的名字。在 **Phase Name** 列下的单元上单击，打开定义名字和统计组的一个窗口。阶段名主要是可读性的需要（第 2 阶段，第 7 事务，认证处理或提交机密信息），对仿真没有影响。如果设置统计组名，那么仿真将为共享同一统计组名的各阶段记录响应时间。响应时间的计算：最后阶段完成其执行的时间与组中第一个阶段开始其执行时的时间之差。

第 2 列 **Start Phase After**（在之后阶段开始）指定阶段将开始的时间。这个属性接受如下值：

1）Application Starts（应用开始）是一个预定义值，如果被选中，则将当前阶段配置为：一旦应用开始，则当前阶段开始。

2）Previous Phase Ends（前一阶段终止）是一个预定义值，如果选择，则意味着当前阶段将在前一阶段（即在上一行中定义的阶段）完成其执行之后立刻开始。

3）可用于这个属性的值的下拉列表也包含迄今为止定义的所有阶段的名字。将 **Start Phase After**（在... 之后阶段开始）列的值设置为一个特定阶段的名字也是可能的，在这种情形中，当指定的阶段完成其执行时，当前阶段将开始。

4）Edit... 指定以逗号分隔的阶段名字的列表。在这样一种情形中，仅当在所提供列表中的所有阶段完成其执行之后，当前阶段才开始。当单个阶段依赖于多个阶段时，这个选项是特别有用的。例如，考虑这样一种状况：其中一个代理服务器要求登录信息和网站的名字，才可允许客户端访问那个站点。当代理服务器接收到所要求的信息时，它就开始认证和名字解析阶段。这些阶段可并发运行，并可以任意顺序完成。仅在认证和名字解析阶段完成之后，数据传输阶段才可开始。在以逗号分隔列表中指定多个阶段名，允许对这种行为进行建模。以逗号分隔列表在逗号和后跟的阶段名之间不应该包含空格。OPNET 在运行时不做任何错误检查，这意味着这个属性将接受任何输入的值，即使一个不正确的值也是如此。但是，在存在错误的情况下（例如，一个不正确的阶段名，或在逗号和下一个列表项之间有一个空格），仿真将不能执行，并将给出一条错误消息 Phase Error: Encountered phase configuration error（阶段错误：遇到阶段配置错误）。

Manual Configuration Table 的第 3 列称作 **Source**（源）。该列指定当前阶段的源节点的一个符号名。对于每个阶段，源节点或目的地节点的实际名字仅解析一次，且任务的所有阶段遵循同一名字解析结果。一个工作站、一台服务器或一个 LAN 对象可作为源节点或目的地节点。这个属性接受如下值：

1）Originating Source（发起源）包括当前任务（即阶段）的一个定制应用，部署于其上的节点将作为当前阶段的源节点。

2）Previous Source（前一个源）在前一阶段过程中作为源的节点，也将作为当前阶段的一个源。

3）Previous Destination（前一目的地）在前一阶段过程中作为目的地的节点，将作为当前阶段的源。

4）Edit... 允许将符号源名指定为一个字符串。这个名字必须通过被仿真网络中一个实际节点的源或目的地首选项属性进行映射或解析。如果 **Source** 符号名是首次在当前任务中被定义的，那么它应该通过相应节点中的 **Source Preferences**（源首选项）属性被映射到实际节点。之后相同的映射将用于整个任务的所有阶段。

下一列被称作 **Destination**（目的地），且它指定当前阶段目的地节点的符号名。这个属性接受如下值：

1）Originating Source（发起源）包括当前任务（即阶段）的一个定制应用，部署于其上的节点，将作为当前阶段的目的地节点。

2）Previous Source（前一个源）在前一阶段过程中作为源的节点，也将作为当前阶段的目的地。

3）Previous Destination（前一目的地）在前一阶段过程中作为目的地的节点，也将作为当前阶段的目的地。

4）Not Applicable（不适用的）在这个阶段中没有目的地节点。典型情况下，在这种情形中，该阶段被用作对源节点处的处理进行建模。同样，如果 **Destination** 属性被设置为 Not Applicable，那么在这个阶段的源和目的地之间定义所发送流量的属性值将被忽略。

5）Edit... 允许将目的地符号名指定为一个字符串。这个名字必须被映射到被仿真网络中的一个实际节点，如果该符号名是首次在任务内使用的话。相同的映射将用于整个任务的所有阶段。

下两列指定在该阶段中产生的流量的特点。列 **Source - > Dest Traffic**（源 - > 目的地流量）定义从源节点到目的地传输的流量的特点。实际上，这个复合属性指定请求消息是如何在源产生的。源—目的地流量特点是通过子属性进行定义的，如图 6.5 所示。

(Source->Dest Traffic) Table

Attribute	Value
Initialization Time (seconds)	constant (0.001)
Request Count	constant (1)
Interrequest Time (seconds)	constant (0)
Request Packet Size (bytes)	constant (1000)
Packets Per Request	constant (1)
Interpacket Time (seconds)	constant (0)
Server Job Name	Not Applicable

Details　Promote　OK　Cancel

图 6.5　源—目的地流量特点的表

1）**Initialization Time**（**seconds**）[初始化时间（秒）] 是在源节点开始产生请求之前逝去的时间。如果为这个阶段没有定义目的地节点，那么这个属性指定在源节点处的处理时间。

2）**Request Count**（请求计数）为从源节点发送到目的地节点的请求数量。

3）**Interrequest Time**（**seconds**）[请求间隔时间（秒）] 是在两个连续的请求之间逝去的时间量。

4）**Request Packet Size**（**bytes**）[请求报文尺寸（字节数）] 是个体报文的尺寸。因为一条请求可由多条报文组成，则这个属性定义了请求之一报文的尺寸。因此，请求本身的尺寸可这样计算：将报文尺寸乘以每条请求的报文数。

5）**Packets Per Request**（每个请求的报文数）指定在单条请求中的报文数。

6）**Interpacket Time**（**seconds**）[报文间隔时间（秒）] 是在属于同一请求的两个连续报文传输之间逝去的时间量。

7）**Server Job Name**（服务器任务名）是与这个阶段相关联的，在 *Server Config*（服务器配置）工具对象内定义的服务器任务名。服务器配置的话题超出了本书的范

围，这里不做讨论。

Manual Configuration Table 的列 **Dest - > Source Traffic**（目的地 - > 源流量）定义从目的地节点到源节点传输流量的特点。实际上，这个复合属性定义目的地节点如何产生响应消息。这个属性又称为 No Response（没有响应）的一个预设值，指明目的地节点不产生响应消息。同样，如果目的地节点的符号名待定，那么这个属性的值就被忽略。通过如下子属性定义目的地—源流量特点，如图 6.6 所示：

(Dest->Source Traffic) Table

Attribute	Value
Request Processing Time (seconds)	constant (0)
Response Packet Size (bytes)	constant (0)
Packets Per Response	constant (0)
Interpacket Time (seconds)	exponential (1.0)
Server Job Name	Not Applicable

Details　Promote　OK　Cancel

图 6.6　目的地—源流量特点的表

1）**Request Processing Time**（**seconds**）［请求处理时间（秒）］指定在目的地节点可产生一条响应之前，它处理请求所花费的时间。

2）**Response Packet Size**（**bytes**）［响应报文尺寸（字节数）］以字节为单位，指定由目的地节点产生的每条响应报文的尺寸（在响应中可存在多条报文）。

3）**Packets Per Response**（每个响应的报文数）作为对请求报文的响应，指定在所产生的每条响应消息中的报文数。

4）**Interpacket Time**（**seconds**）［报文间隔时间（秒）］是在属于同一响应消息的两次连续报文传输之间逝去的时间量。

5）**Server Job Name**（服务器任务名）是与这个阶段相关联的服务器任务名。服务器配置的话题超出了本书的范围，这里不做讨论。

Manual Configuration Table 的下一列是 **REQ/RESP Pattern**（REQ/RESP 模式）。它指定在源节点和目的地节点之间消息交换模式。这个属性仅支持两个预设值：

1）REQ - > RESP - > REQ - > RESP...［Serial（串行的）］值，指明除非源从目的地接收到一条响应，否则源将不产生一条新请求。具体而言，在响应消息到达之后发送下一条请求消息之前，源节点将等待 **Interrequest Time**（**seconds**）［请求间隔时间（秒）］。如果 **Dest - > Source Traffic**（目的地 - > 源流量）属性被设置为 No Response（无响应），则这个值是不可用的。

2）REQ - > REQ - > ... - > RESP...［Concurrent（并行的）］值，指明在不等待响应到达的情况下，依据 **Interrequest Time**（**seconds**）［请求间隔时间（秒）］属性的配置，由源产生请求。实际上，请求消息将总是以 **Interrequest Time**（**seconds**）为间

隔产生，独立于目的地响应。

Manual Configuration Table 的下一列 **End Phase When**（终止阶段的时间），指定表示一个阶段完成的条件。这个属性允许如下值之一：

1）Final Request Leaves Source（最终请求离开源节点）——在源节点产生 **Request Count** 个请求消息之后，该阶段完成（源已经发送最后请求的最后报文）。

2）Final Request Arrives at Destination（最终请求到达目的地节点）——在目的地节点接收到 **Request Count** 条请求消息之后，该阶段完成（最后请求的最后报文已经到达目的地节点）。

3）Final Response Leave Destination（最终响应离开目的地）——在目的地节点对来自源的最后请求产生响应消息之后，该阶段完成（即目的地已经发送最后响应消息的最后报文）。

4）Final Response Arrives at Source（最终响应到达源节点）——在源节点接收到来自目的地的最后响应消息之后，该阶段完成（即最后响应消息的最后报文到达源节点）。

列 **Timeout Properties**（超时性质）指明允许该阶段执行的最大时间。添加这个属性是为了处理这样的情况，其中阶段连接被终止，或阶段事务报文已经丢失。如果阶段没有在指定时间内完成，那么它就被强制终止，不管所有事务是否结束。这个属性也允许指定在一次强制阶段终止之后将执行什么操作：下一阶段（即继续正常任务执行）或下一任务（即忽略本任务的剩余阶段）。这个属性有称作 Not Used（未使用）的一个预设值，指明在终止之前，该阶段必须完成它的所有事务。

Manual Configuration Table 的最后一列称为 **Transport Connection**（传输连接），且它指定这个阶段如何使用传输协议。图6.7形象地说明了 *Transport Connection Table*，它由两个子属性组成：

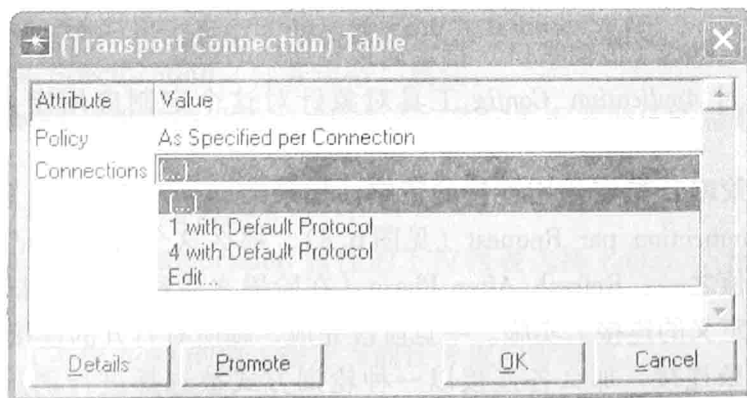


图6.7　*Transport Connection Table*（传输连接表）

1）**Policy**（策略）定义在这个阶段中如何建立传输连接。这个属性接受两个值：

① New Connection per Request（每个请求都采用新连接）意味着从源到目的地的

请求将在一条独立的连接上传输。在源接收到响应后，相应的连接被关闭。对每条连接仅允许一个请求—响应对。

② As Specified per Connection（每个连接依据指定的进行）意味着连接策略是通过称作 **Connections**（连接）的其他复合属性确定的。

2）**Connections**（连接）是一个复合属性，指定如下信息，如连接数、传输协议、传输端口和连接重用的策略。图 6.8 形象地说明了 *Connections Table*（连接表）。每个阶段的连接数是通过标题为 *Rows* 的文本框进行控制的。本表中的其他按钮提供了其他类似表（如 *Manual Configuration Table*）的相同功能。*Connections Table* 中子属性的含义如下：

① **Tag**（标签）是用于识别共享连接的连接符号名。

② **Transport Protocol**（传输协议）是这条连接使用的传输协议名。如果设置为 Default，那么这条连接将在通过 *Application Config* 工具对象针对这个定制应用定义的传输协议之上，运行这条连接。

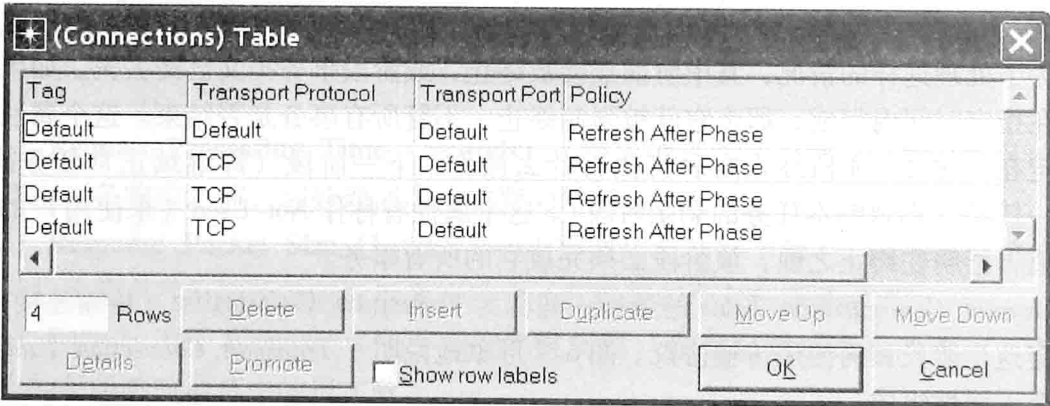


图 6.8　*Connections Table*（连接表）

③ **Transport Port**（传输端口）指定连接将连接的端口号。如果设置为 Default，那么这条连接将通过 *Application Config* 工具对象针对这个定制应用定义的端口号进行连接。

④ **Policy**（策略）指定重用连接的策略。如果 *Transport Connect Table* 的 **Policy** 属性设置为 New Connection per Request（见图 6.8），那么这个属性的配置将被忽略。这个属性可取两个值之一：Refresh After Phase（在阶段之后刷新）意味着这个阶段的所有消息传递将在定义的连接上实施。一旦阶段完成，则所有打开的连接将被关闭。如果该阶段配置有多条连接，那么各连接以一种轮询方式被选择进行消息传递。而 Reuse Across Phases（在阶段间重用）意味着连接属于整个任务，且重用连接的实际策略通过 *Task Config* 对象的 **Connection Policy** 属性（见图 6.3）进行定义。一旦阶段完成，则直到到达任务的最后阶段之前，连接将保持打开状态。在此之后，连接将是关闭的［如果 **Connection Policy**（连接策略）属性值被设置为 Refresh After Task（在任务之后刷

新)] 或被重用 [如果连接策略属性值被设置为 Reuse Across Tasks (在任务间重用)]。具体而言，在后一种情形中，直到该任务所属的应用终止之前，连接都将保持打开状态。

注意仅在同一应用实例间才允许连接共享。例如，如果两个节点执行同一应用，那么每个节点创建一个新的应用实例，则因此连接不能在这些应用间共享。在实践中，如果传输流量的一个新节点要启动，那么它首先搜索具有相同连接标签值的任何共享连接。如果找到这样的一条连接，那么就重用那条连接，否则，发起一条新的连接。一旦阶段完成，则它检查 *Connections Table* (连接表) (如 Reuse Across Phases 或 Refresh After Phase) 中 **Policy** 属性的值来确定该连接是否要终止。上述情况不适用于这样的情形，此时 *Transport Connection Table* (传输连接表) 中的 **Policy** 属性被设置为 New Connection Per Request (每个请求采用新连接)，其中为阶段内的每条连接打开一条新的连接。

Transport Connection Table 中的 **Connections** 属性接受如下两个预设值之一：

1) 1 with Default Protocol (1 条连接，采用默认协议) ——配置单个默认传输协议，连接到一个默认端口，在本阶段之后刷新连接。

2) 4 with Default Protocol (4 条连接，采用默认协议) ——总共配置四个默认协议。第一条连接运行在默认传输协议之上，连接到默认端口，且在本阶段之后被刷新。其他三条连接是相同的，例外情况是，它们是显式配置为在 TCP 上运行的。这样的一种配置如图 6.8 所示。

6.2.4　小结：为定制应用配置任务

6.2.2 节和 6.2.3 节描述了用来为定制应用定义而配置任务和阶段的各种属性。通过为一个定制应用指定一项任务提供各步骤的小结结束本节：

1) 将 *Task Config* 对象添加到仿真场景的工作空间。

2) 在 *Task Config* 对象上右击，并选择 **Edit Attributes** 选项。

3) 展开 **Task Specification** (任务规格) 属性。

4) 通过将 **Rows** 属性的值设置为相应的值，指定要配置的任务总数。

5) 对于每行：

① 展开行，并设置任务的名字。

② 通过从 **Manual Configuration** 属性的下拉列表选择 Edit...，编辑 *Manual Configuration Table*。

- 通过设置文本框 *Rows* 的值，指定当前任务重的阶段数。
- 配置任务的每个个体阶段。
- 当完成任务的所有阶段时，单击 **OK** 按钮保存阶段配置。

③ 设置 **Connection Policy** 属性的值。

6) 当配置完成所有的任务时，单击 **OK** 按钮保持改变，并退出 *Task Config* **Edit Attributes** 窗口。

6.3 在 OPNET 中定义定制应用

在指定一项定制应用的所有任务之后，通过 *Application Config* 对象，定义定制应用本身。定义定制应用的过程与标准应用的非常类似，且它由如下步骤组成：

1）添加 *Application Config* 对象（如果它还没有在项目工作空间中的话）。

2）右击 *Application Config* 对象，打开 **Edit Attributes** 窗口。

3）展开 *Application Definitions* 复合属性，并指定要配置的应用总数（即设置 **Number of Rows** 属性的值）。

4）一次展开一行，配置每项应用：

① 设置 **Name** 属性的值。

② 展开 **Description** 属性，并配置 *Custom* 应用。

5）为其他应用，重复上述过程。

6）单击 **OK** 按钮关闭窗口，并保存改变。

在这些步骤与标准应用的那些步骤（见 5.3.2 节）之间的唯一差异是可用于定义 *Custom* 应用的属性集合，如图 6.9 所示。

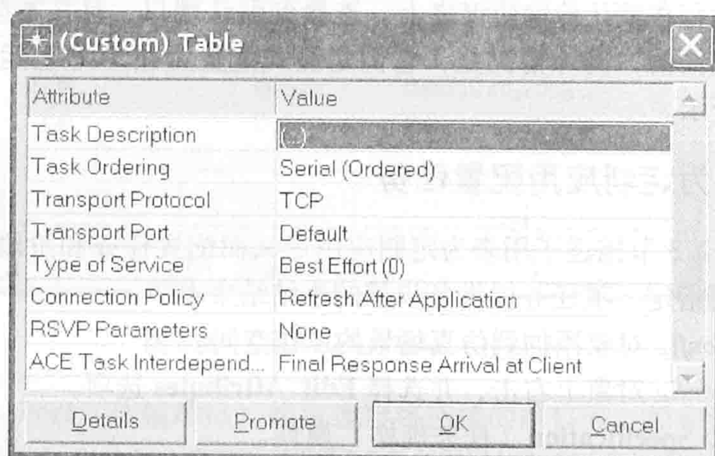


图 6.9 可用于定义定制应用的属性

不像标准应用的是，定制应用的定义不包括诸如报文尺寸和数据速率等的信息。所有这些属性已经在相应的任务和阶段内进行定义。一项定制应用的定义主要处理指定要使用哪些任务以及这些任务如何执行。具体而言，*Custom* 应用定义由如下属性组成：

1）**Task Description**（任务描述）定义在这项定制应用中要用的任务。这个复合属性包括两个子属性：

① **Task Name**（任务名字）指定任务的名字。仅有在 *Task Config* 对象中已经定义的那些任务才可被选择作为这个属性的值。事实上，这个属性的下拉菜单提供可用任务的列表。

② **Task Weight**（任务权重）控制当前任务将被选择的概率。仅当在应用内存多项

任务且各任务被配置为以一种随机方式运行时，才可使用这个属性。在这样一种情形中，一项任务的选择概率是这样计算的：作为该任务的权重与这个应用中所有任务之权重总和的比率。

2）**Task Ordering**（任务排序）指定任务执行的顺序。这个属性可被设置为三个预设值之一：Serial（Ordered）[顺序（有序的）]、Serial（Random）[顺序（随机的）]或Concurrent（并行的）。每个这样的值都被配置为在定制应用内执行所有的任务，其中使用对应于值的名字的排序方案。这些值的含义是自解释的。这个复合属性也由两个子属性组成，每个都有相同的名字：

① **Task Ordering**（任务排序）子属性也可设置为Serial（Ordered）[顺序（有序的）]、Serial（Random）[顺序（随机的）]或Concurrent（并行的）。注意，如果父属性**Task Ordering**被设置为其值之一，那么其子属性**Task Ordering**将自动地设置为具有相同名字的相应值。

② **Number of Used Tasks**（所用任务的数量）指定将执行的应用任务数。这个属性的默认值是Number of Specified Tasks（所指定任务数），这意味着仿真将尝试执行为这个应用定义的所有任务。如果这个属性值被设置为一个数，如N，那么仅有应用内的N个任务将被执行。具体而言，如果选择Serial（Ordered）或Concurrent任务排序，那么仅有前N个应用任务将被执行。但是，如果应用被配置为使用Serial（Random）排序运行各项任务，那么仿真将随机选择N个任务。

③ **Transport Protocol**（传输协议）指定定制应用所用的传输协议。仅当应用在任务的阶段使其**Transport Protocol**属性值设置为Default时，才可使用这个值。否则，为该应用阶段定义的传输协议将优先使用。这个属性的预定义值之一是Direct Delivery（直接交付）。如果一个定制应用的传输协议被设置为Direct Delivery，那么在应用实例之间直接仿真消息交换，且忽略与传输、网络、媒介访问和其他层相关联的所有延迟和其他特征。

④ **Transport Port**（传输端口）指定定制应用使用的端口号。仅当应用在任务的阶段使其**Transport Port**属性值设置为Default时，才可使用这个值。否则，为该应用阶段定义的端口号将优先使用。

⑤ **Type of Service**（服务类型）指定IP报文首部中*ToS/DiffServ*字段的值。欲了解更多细节，参见5.4.1节。

⑥ **Connection Policy**（连接策略）指定连接重用策略。这个属性仅可被设置为值Refresh After Application（在应用之后刷新），这意味着在应用完成时终止连接，且在Task Config对象中定义的连接重用策略监测应用的消息交换。

⑦ **RSVP Parameters**（RSVP参数）指定RSVP的配置。欲了解更多细节，参见5.4.1节。

⑧ **ACE Task Interdependency**（ACE任务相互依赖关系）使用ACE处理定制应用的定义。这个话题超出了本书的范围，这里不做讨论。

6.4 在 OPNET 中配置定制应用的例子

这里使用图 6.10 所示的例子，形象地说明在 OPNET 中配置定制应用的过程。这是与 6.1 节中讨论的例子相同的一个例子。如图 6.10 所示的定制应用可被配置为具有几个阶段的单个任务。但是，为了形象地说明 OPNET 中存在的其他功能特征，这里将这个应用建模为使用三个独立任务，它们具有如下各阶段。

1. 登录任务

1）提交机密信息阶段：客户端提交其机密信息（尺寸为 1024B 的单条消息）。Web 服务器在与认证服务器验证机密信息之后，才将一条响应发送给客户端。

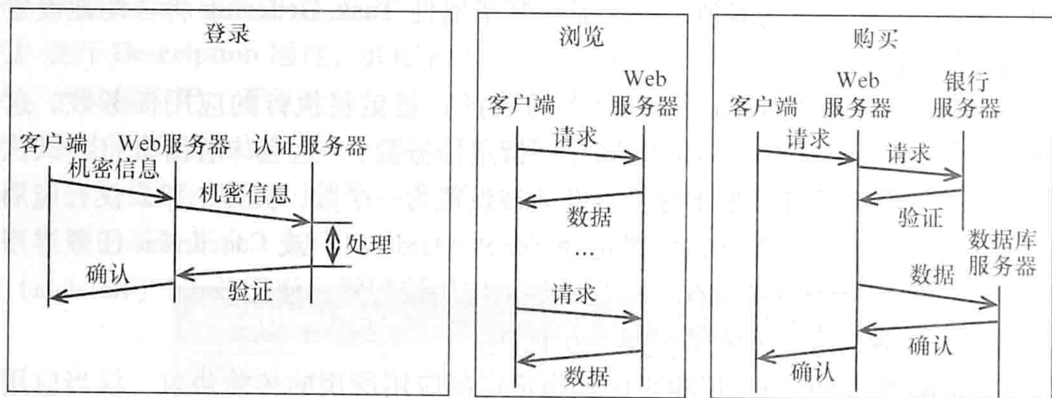


图 6.10 一个多层应用的例子

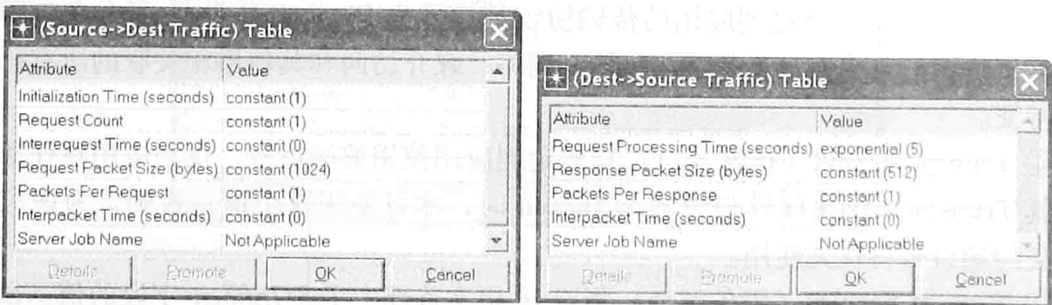


图 6.11 Verify Credentials（验证机密信息）阶段的流量配置

2）验证机密信息阶段：Web 服务器将客户端的机密信息（单条 1024B 的消息）转发给认证服务器，在发送响应（由单条 512B 的报文组成）之前处理数据要花费约 5s。图 6.11 给出这个阶段两个方向的流量配置。注意仿真在认证服务器处的处理方法是，将 **Request Processing Time（seconds）**［请求处理时间（秒）］属性的值设置为 5s。

3）发送 *ACK* 阶段：在 Web 服务器从认证服务器接收到验证之后，它将一条 ACK 发回客户端。ACK 消息是 1024 字节。

2. 浏览任务

浏览阶段：客户端周期性地将一条请求提交到 Web 服务器。每条客户端请求是 1024B，并由单条消息组成。在客户端发出另一条请求之前，花费 5min 处理接收到的信息。在购买一件物品之前，客户端发送大约 40 条请求。对于每条请求，Web 服务器发送长度为 10 000B 的一条响应消息。Web 服务器花费大约 2s 检索有关被请求物品的信息。图 6.12 形象地说明这个阶段两个方向的流量配置。产生请求消息的请求间隔时间被设置为 300s，这代表 5min，客户端花费在处理从 Web 服务器来的信息的时间。响应报文尺寸被设置为 2048B。因为每条响应消息有 5 条报文，则响应的总尺寸为 5 × 2048B = 10 000B。

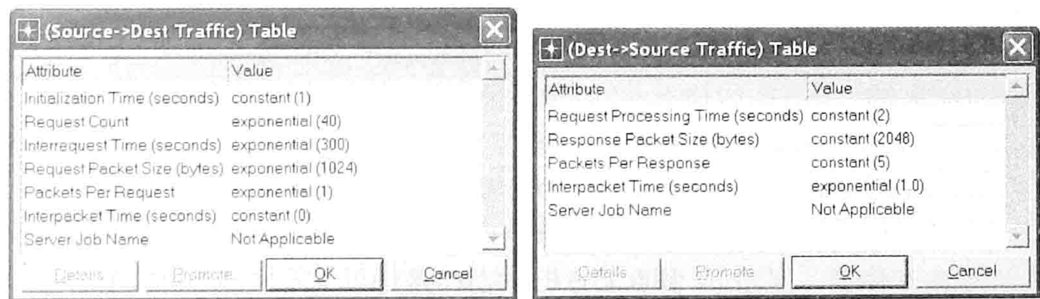


图 6.12　浏览阶段的流量配置

3. 购买任务：

1）提交请求阶段：客户端将一条请求发送到 Web 服务器，购买一件物品。请求是 6000B，并被分割为 3 条消息。直到 Web 服务器从其他服务器接收到一条 ACK 之前，它不发送一条响应。

2）验证 *CC* 阶段：Web 服务器将客户端的信用卡信息发送到银行服务器。信用卡信息作为单条 2000B 的报文进行发送。银行服务器验证所提供信息的正确性，并发回带有一个确认码的一条响应消息。响应消息的尺寸为 512B。

3）更新数据库阶段：这个阶段与验证 *CC* 阶段同时开始。Web 服务器将有关所购物品的信息发到数据库服务器。请求尺寸是 2000B，且它由单条消息组成。响应报文的尺寸为 128B。

4）完成购买阶段：一旦 Web 服务器从银行服务器和数据库服务器接收到响应消息（即验证 *CC* 和更新数据库阶段都已经完成），则它产生发回到客户端的一条消息。确认消息为 10000B，且它被分割为 5 条报文。

由于篇幅考虑，上述例子没有对如下情况进行建模，例如不能登录（即无效的登录信息）和不能购买（即无效的信用卡信息）。在图 6.13 中给出任务配置的汇总。下面是有关上述配置的一些观察结果：

1）出于简单性考虑，保持传输连接策略设置为默认值。

2）客户端节点被表示为 *Originating Source*（发起源），是定义应用所部署的节点。

3）登录任务的提交机密信息和发送 *ACK* 阶段没有目的地产生的响应。这些阶段代

表由客户端发送到 Web 服务器的请求消息和在 Web 服务器完成机密信息的验证之后发送的一条响应。

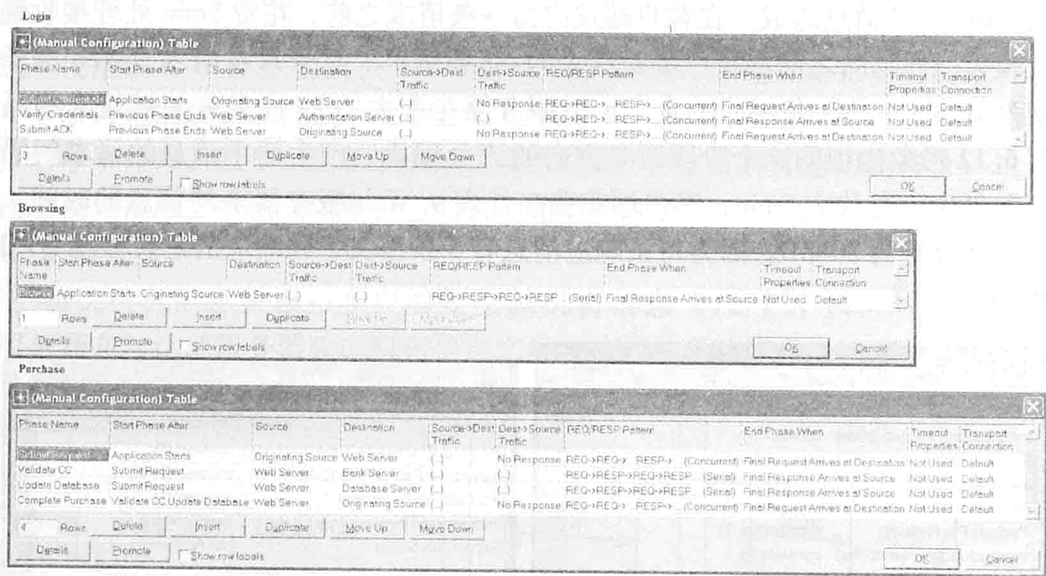


图 6.13　任务配置的汇总

● 验证机密信息阶段被配置为具有并行的 REQ/RESP 模式。但是，验证机密信息阶段的这个属性的实际设置是无关的，原因是源产生单条请求消息。

● 浏览阶段被配置为具有串行 REQ/RESP 模式，因为在客户端接收到其前一消息的响应之前，它不应该产生一条新的请求。

● 在购买任务中，在提交请求阶段之后，开始验证 *CC* 和更新数据库阶段。这种配置对这些阶段的并行执行进行建模。结果是，完成购买阶段被配置为仅在验证 *CC* 和更新数据库阶段完成之后才开始。

一旦指定了所有的阶段，留下的仅有事情是定义定制应用本身。图 6.14 形象地说明了在这个例子中描述的定制应用的配置。这个应用配置为以串行顺序执行任务：登录、浏览和购买。因为任务的执行顺序被配置为串行的，所以任务的权重值是无关的。

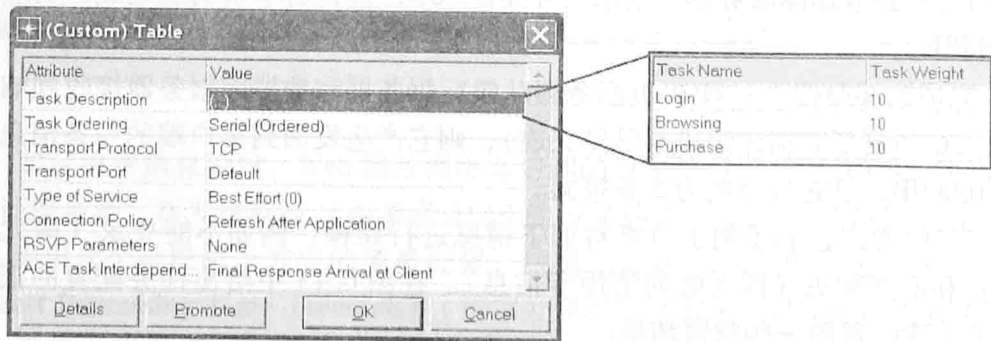


图 6.14　定制应用的配置

6.5　显式报文产生源

显式报文产生是建模流量源的另一种方法。如第 5 章所述，这种技术不能对任何特定的应用层协议进行建模。相反，它仿真由一个端节点产生逐条报文的流量。仅有在其名字中包含单词_ *station* 的端节点模型才包括定义显式报文产生源的属性。但是，不同的节点模型经常以不同方式定义报文产生参数。

例如，如图 6. 15 中的 *ethernet_ ip_ station_ adv* 模型，在称为 **Traffic Generation Parameters**（流量产生参数）的一个复合属性之下定义报文产生参数，该属性是复合属性 **IP** 的一个子属性。这个模型通过诸如 **Packet Inter - Arrival Time**（**seconds**）［报文到达间隔时间（秒）］、**Packet Size**（**bytes**）［报文尺寸（字节）］、**Destination IP Address**（目的地 IP 地址）、**Start Time**（**seconds**）［开始时间（秒）］和 **Type of Service**（服务类型）等属性指定报文产生配置。这里不详细描述这些属性，原因是这些属性的目标是自解释的。注意属性 **Destination IP Address** 要求目的地节点的一个实际 IP 地址。默认情况下，被仿真网络系统中的所有节点是在仿真执行过程中动态地被指派 IP 地址的。但是，在仿真执行之前，可设置网络中任意节点的 IP 地址。在端节点中，IP 地址是通过属性 **IP... IP Host Parameters... Interface Information... Address**（IP... IP 主机参数... 接口信息... 地址）进行配置的。另外一种方法是，通过从 **Project Editor**（见第 9 章）的下拉菜单中选择选项 **Protocols→IP→Addressing→ Auto - Assign IPv4 Addresses**（协议→IP→寻址→自动指派 IPv4 地址），可为网络中的所有节点设置 IP 地

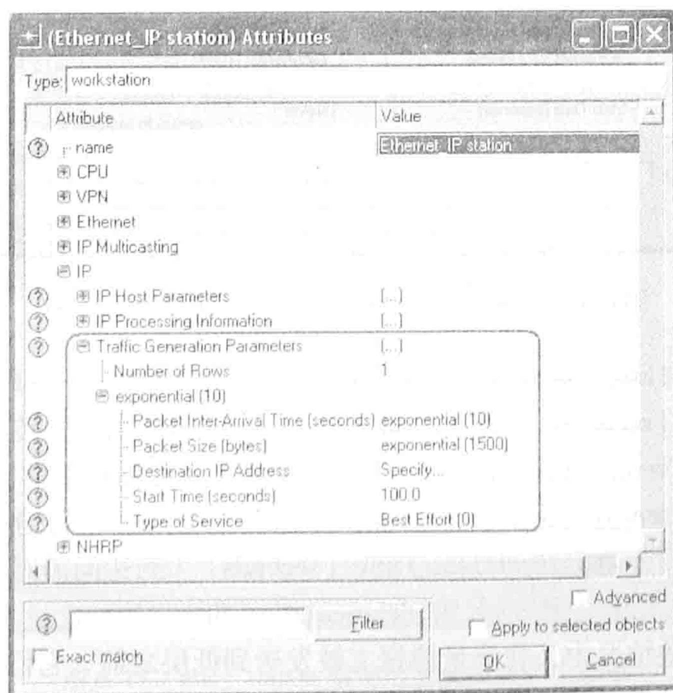


图 6. 15　*ethernet_ ip_ station_ adv* 模型的显式报文产生参数

址。一旦目的地端节点被指派 IP 地址，就需要检索那些值，方法是检查每个节点的 **IP... IP Host Parameters... Interface Information... Address** 属性的值，之后在显式产生报文的端节点中将值指派给 **Destination IP Address**（目的地 IP 地址）。

在一个节点配置的流量源定义，仅指定从这个节点到一个目的地的一个方向中的流量。为了配置反方向的流量，也要求在目的地节点处定义流量源。从这个节点发起的流量源数量，是通过属性 **Number of Rows** 加以控制的，其中每行代表这个节点和一个目的地之间的单条流量断续流。

另外，如图 6.16 所示的 *ethcoax_ station* 模型，通过顶层复合属性 **Traffic Generation Parameters**（流量产生参数）定义 ON - OFF 流量源。ON - OFF 流量源模型由一系列 ON - OFF 时段（即 ON、OFF、ON、OFF、ON、OFF 等）组成，其中源在 ON 时段过程中产生流量，在 OFF 时段是空闲的。**Traffic Generation Parameters** 属性由如下子属性组成：

1） **Start Time**（**seconds**）［开始时间（秒）］——当节点开始传输流量的时间。

2） **ON State Time**（**seconds**）［ON 状态时间（秒）］——ON 时段的时长。

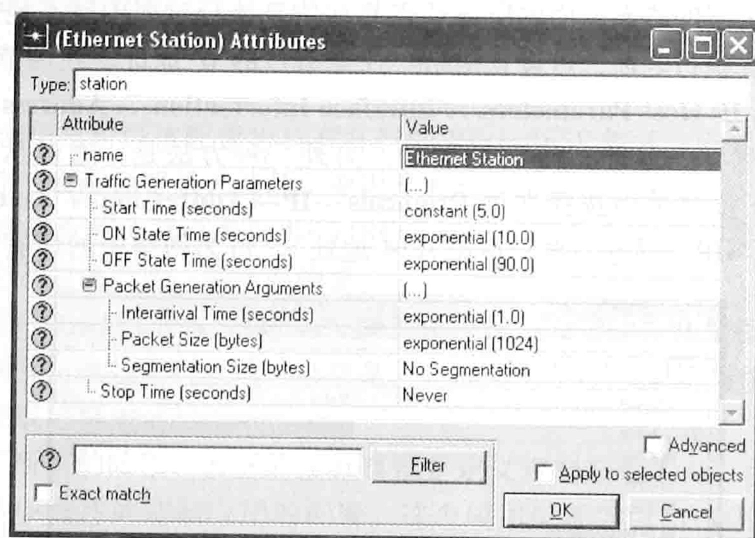


图 6.16 *ethcoax_ station* 模型的显式报文产生参数

3） **OFF State Time**（**seconds**）［OFF 状态时间（秒）］——OFF 时段的时长。

4） **Stop Time**（**seconds**）［停止时间（秒）］——当节点停止传输流量的时间。这个属性接受预设值 Never，它将节点配置为直到仿真结束之前都传输流量。

5） **Packet Generation Arguments**（报文产生参数）是一个复合属性，指定所产生流量的特征。这个属性由 **Interarrival Time**（**seconds**）［到达间隔时间（秒）］、**Packet Size**（**bytes**）［报文尺寸（字节）］和 **Segmentation Size**（**bytes**）［分段尺寸（字节）］子属性组成。在某些情况中，在流量源报文被发送到低层之前，它们被分段为较小的数据块。属性 **Segmentation Size**（**bytes**）指定这样一个分段的尺寸。如果这个属性被设置为值 No Segmentation（不分段），那么在不分段的情况下，报文被发送到低层。

在 *atm_ station* 模型中的报文产生是以另外一种方式定义的。ATM 节点模型包含称为 **ATM Application Parameters**（ATM 应用参数）的一个复合属性，由如下子属性组成：**Arrival Parameters**（到达参数）、**AAL Parameters**（AAL 参数）、**Traffic Contract**（流量契约）和 **Destination Address**（目的地地址），如图 6.17 所示。实际流量特征是通过复合属性 **Arrival Parameters** 定义的，该属性包括子属性 **Packet Interarrival Time**（**seconds**）［报文到达间隔时间（秒）］、**Packet Size**（**bytes**）［报文尺寸（字节）］、**Call Start Time**（**seconds**）［呼叫到达时间（秒）］、**Call Duration**（**seconds**）［呼叫时长（秒）］、**Transmission Size**（**Mbytes**）［传输尺寸（M 字节）］（在单个呼叫过程中传输的总流量）和 **Call Interarrival Time**（**seconds**）［呼叫到达间隔时间（秒）］。另外，在其名字中包括扩展名 *uni_ src* 的 ATM 节点模型也支持报文产生流量源的配置。

除了这些差异外，配置显式报文产生是十分直接的。首先，需要确保关注的节点实际上支持显式报文产生流量。下一步是识别配置显式报文产生的属性的位置。典型情况下，这种属性被归组在一个称为 **Traffic Generation Parameters**（流量产生参数）、**Application Parameters**（应用参数）或一个类似名字的复合属性之下。最后一步是指定定义显式报文产生源的特征的属性值。典型情况下，这样的流量源是通过如下参数定义的，这些参数如指定报文产生开始和结束的时间、报文到达间隔时间、报文尺寸和目的地节点的地址。

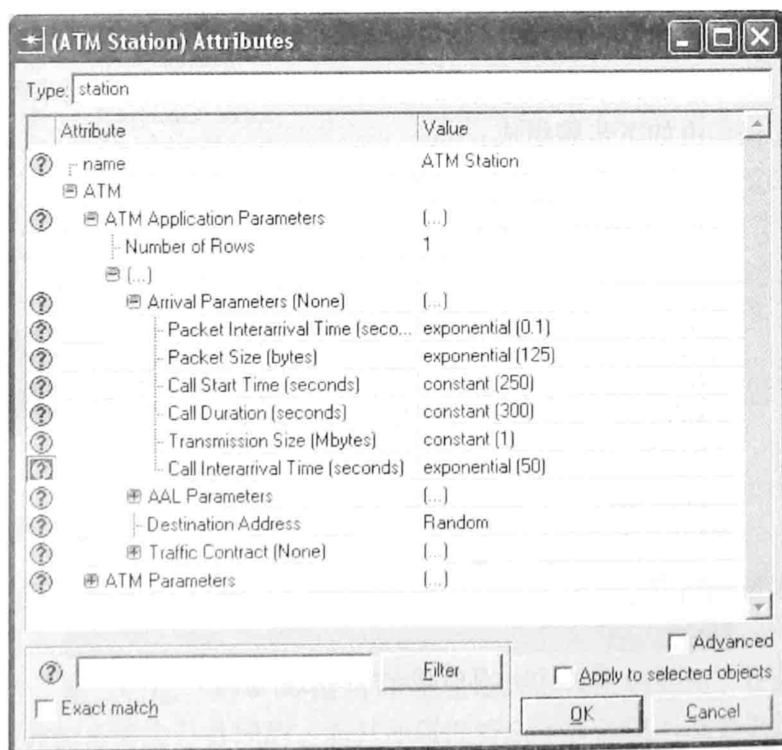


图 6.17　*atm_ station* 模型的显式报文产生参数

6.6 应用需求和流量断续流

Demand（需求）是指定网络中两个节点间流量断续流的另一种机制。OPNET 在应用需求和流量断续流之间做出区分。一个 **application demand**（应用需求）是一个需求对象，它对两个端节点之间的应用层消息交换进行建模。

Traffic flow（流量断续流）是需求的另一种类型，在不表示流量断续流模式细节的情况下，仿真端到端流量。流量断续流需求仅指定在网络处或更具体而言的 IP 层产生的流量。仅有包含 IP 层的节点才能够部署流量断续流的需求。每个需求模型具有一个不同的配置属性集合，它们主要依赖于被建模的流量类型。需求仅表示某种类型流量的特征，它们不对任何具体协议建模。需求是在两个节点之间定义的，这两个节点未必通过一条直接链路连接。由仿真确定需求流量是如何通过被仿真的网络进行路由的。描述所有可用的流量需求模型会花费许多时间，并就这本身而言就需要一个独立章节才行。那也就是仅给出应用和流量断续流需求之特征的概要描述的原因。如果多少熟悉被建模流量类型（如 IP ping、VoIP 和 ATM PVX）的性质，那么应该能够容易地区分相应需求属性的含义。

在一个被仿真系统中指定需求的过程，类似于在一个网络拓扑中以一条链路连接节点的过程。唯一的区别是链路连接两个直接连接的节点，而需求连接网络中的任意两个节点。图 6.18 形象地说明了需求和链路之间的差异。典型情况下，一个需求对象表示为一个点画线，而一条链路表示为网络拓扑中的一条实线。将一个需求对象添加到一个仿真场景中的过程，由如下步骤组成：

1）识别 **Object Palette Tree** 中关注的需求模型。

2）将需求模型拖入 **Project Editor**。

3）连接任意两个期望的节点。

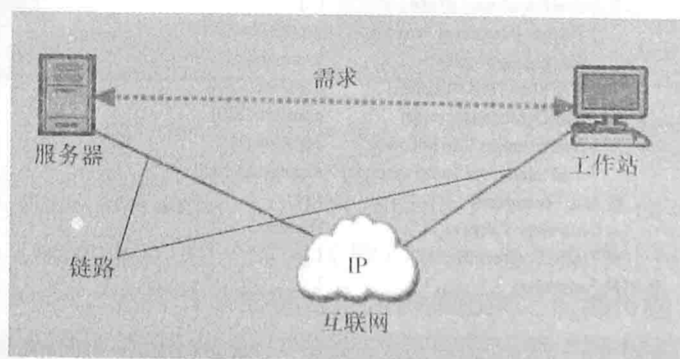


图 6.18 具有需求的网络拓扑例子

6.6.1 应用需求

应用需求对源节点和目的节点之间以两个方向流动的应用流量进行建模。默认情况下，应用需求表示为点画线，在两端有箭头指明双向流量。如图 6.19 所示，通过如下

属性配置应用需求：

1）**Duration**（时长）是一个复合属性，指定应用需求的开始时间和结束时间。

2）**Request Parameters**（请求参数）是一个复合参数，指定每条请求消息的尺寸、请求消息的产生速率（即每小时产生的请求数）和每条请求报文的 ToS 值。

3）**Response Parameters**（响应参数）是一个复合属性，由单个子属性 **Size (bytes)**［尺寸（字节）］组成，它指定响应消息的尺寸。

4）**Traffic Mix**（%）［混合流量（%）］指定将如何建模应用需求。特别地，它定义所建模的需求流量为背景流量的百分比。这个属性可取几个预设值中的任何一个：All Discrete（都是离散的）、25%、50%、75%、All Background（都是背景流量）和 Edit...。All Discrete 设置对应于 0% 的值，而 All Background 对应于 100% 的一个值。设置 Edit... 可输入任意期望的值。

5）**Transport Protocol**（传输协议）指定在其上要传输的应用需求的请求和响应消息的传输协议。

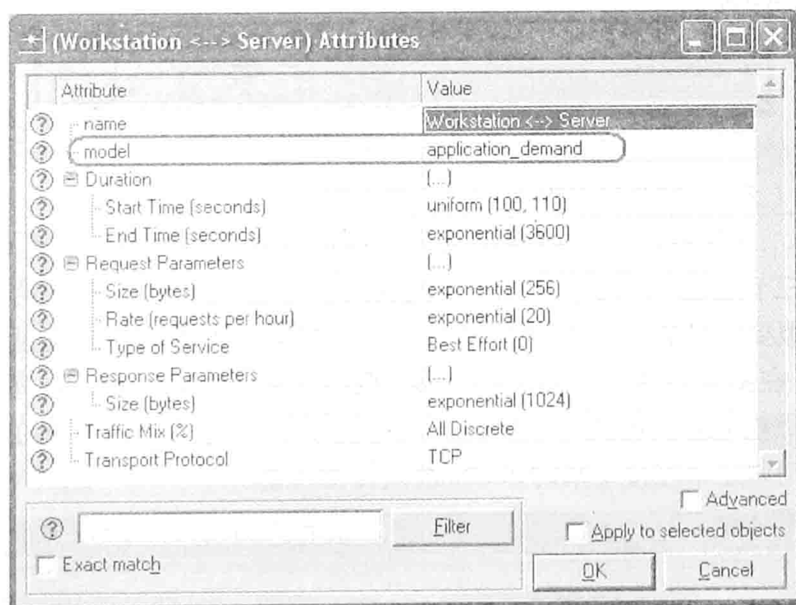


图 6.19　应用需求对象的属性

6.6.2　流量断续流需求

流量断续流对某种类型的流量进行建模，这种流量可以是单向的或双向的。建模单向断续流的流量断续流表示为一端有一个箭头的一条点画线。流量断续流方向的箭头，是节点所指向的，是目的地。图 6.20 给出 IP 流量断续流需求的属性。在这幅图中以圆圈标出的属性对多数流量需求是共同的，而其他属性特定于要建模的流量断续流的类型。

属性 **Traffic (bits/second)**［流量（比特/秒）］和 **Traffic (packets/second)**［流量（报文数/秒）］指定在时段上流量断续流改变的传输速率。这些属性也称作概要，且它们有如下可能的值：

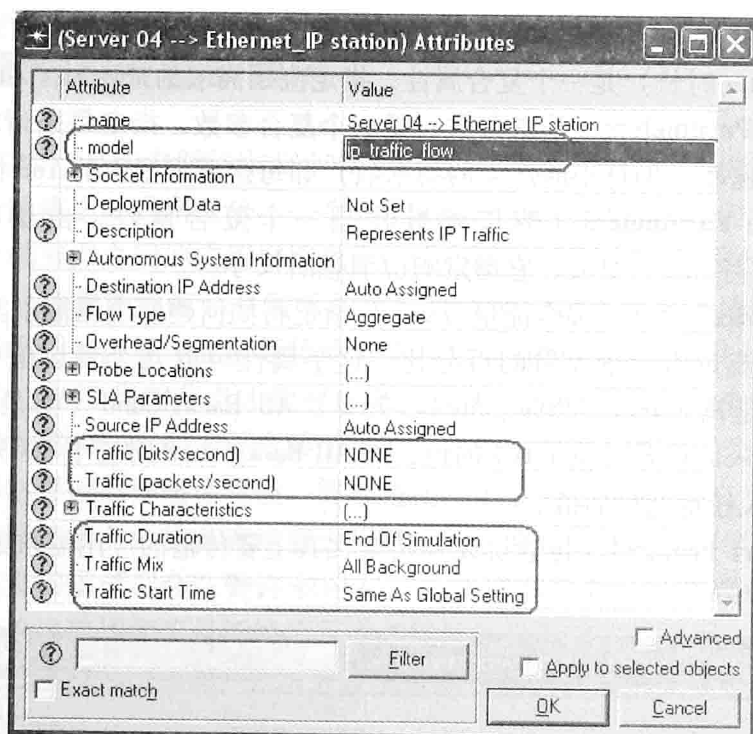


图 6.20　IP 流量断续流需求的属性

1）NONE（无），这意味着没有定义概要。

2）Select...（选择...），提供可从中选择的流量概要列表。如图 6.21 所示，一个流量概要由三节组成：*Profile Library*（概要库，指定概要定义的位置）、*Profile*（概要，提供可用概要定义的一个列表）和 *Preview*（预览，将所选中的流量概要表示为一幅图）。例如，在图 6.21 中，*T3_ 1hour_ bps* 概要是从 *demand_ profiles* 库选择的，如预览窗口所示，所选中的概要以 44.736Mbit/s（即 T3 线的速率）的速率在时间 0 开始传输流量，并以相同速率在接下来的 3600s 继续传输流量。

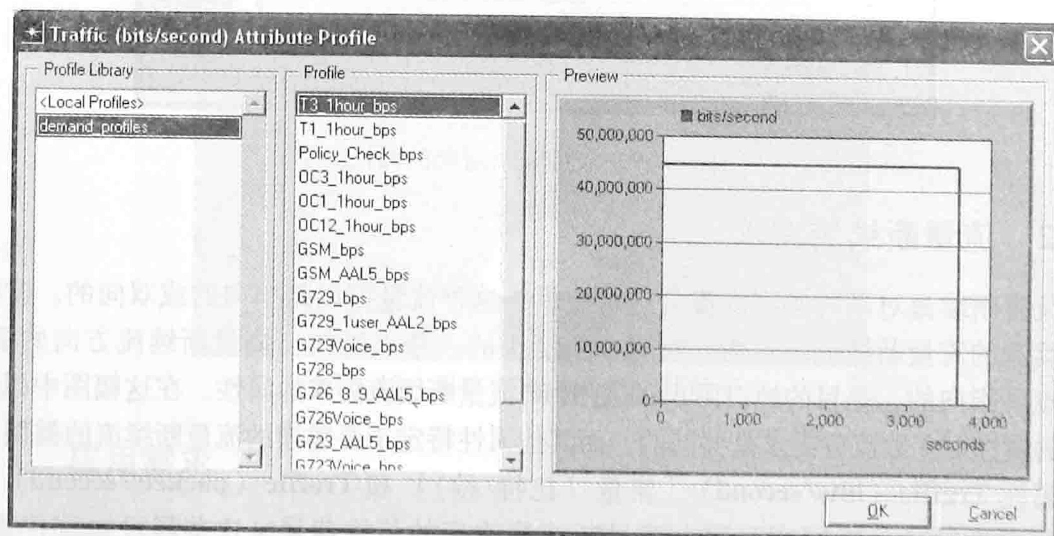


图 6.21　选择预设流量概要

在属性的值字段中选择 Edit...，打开一个窗口，其中可定义一个新的流量概要，如图 6.22 所示。实际流量概要定义是通过有两列的一个表指定的：*seconds*（即 x 轴）和 *bits/second*（比特/秒）或 *packets/second*（报文数/秒）（即 y 轴），这取决于被选择的流量概要的类型。图 6.22 给出属性 **Traffic（packets/second）**［流量（报文数/秒）］的概要配置，这就是表中的第 2 列标题为 *packets/second* 的原因。表本身指定了流量概要的传输速率如何随时间变化。表中的每个表项指示一个新的传输速率的开始。例如，如图 6.22 所示，在时刻 300s 处，流量断续流的传输速率设置为 20 个报文数/s，且在时刻 600s 处，传输速率改变为 5 个报文数/s，这意味着从时刻 300s 到时刻 600s，流量断续流将以 20 个报文数/s 的速率传输数据，而在时刻 600s 处，速率将降低到 5 个报文数/s。流量断续流将继续以 5 个报文数/s 的速率在额外的 100s 期间传输数据，直到时刻 700s 时流量断续流速率改变为 15 个报文数/s。表右侧的预览窗口提供传输速率变化的一个图形表示。

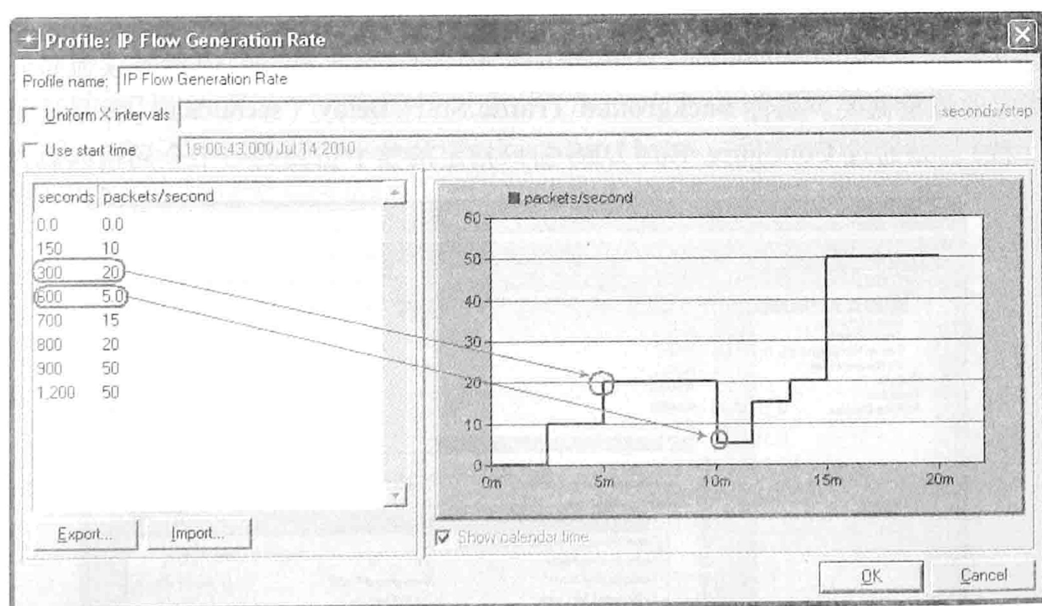


图 6.22　定义流量概要

流量概要窗口的顶部有一个标题为 *Profile Name*（概要名）的文本框，这使您可为给定流量概要定义一个名字。在其正下方是一个检查框 *Uniform X intervals*（均匀的 X 间隔），当选中时，强制定义流量断续流的传输速率的数据点为均匀分布的。x 轴更新间隔的长度可通过相应的文本框加以指定。例如，如果 x 轴更新间隔被设置为 100s，那么流量断续流的传输速率将被定义为每 100s（更新一次）。但是，如果这个检查框没有被选择，那么流量断续流的传输速率可以如图 6.22 所示的任意长度的间隔加以指定，图 6.22 中前两个间隔的长度为 150s，第二个间隔的长度是 300s，而第四个间隔仅有 100s 的长度（即 700 - 600 = 100s）。另一个检查框 *Use Start Time*（使用开始时间）指定概要将开始的时间。最后，OPNET 支持输出和输入流量概要。当单击流量概要窗口左下角的按钮 **Export...**（输出...）时，所定义的流量概要被输出到一个以 tab 隔开的文本

文件，该文件可以任意文本或表格（spreadsheet）编辑器打开。单击 **Import...**（输入...）按钮支持从一个 tab 隔开的文本文件输入一个定义好的流量概要。

继续对图 6.20 所示流量断续流需求属性的讨论，复合属性 Traffic Characteristics（流量特征）包含如下配置参数，如报文尺寸、报文到达间隔时间、报文标记和其他描述这个需求所产生报文的性质的参数（图 6.20 没有给出这些配置参数）。即使这个属性对许多流量断续流定义是共同的，描述报文性质的实际配置参数也经常随流量类型而改变。属性 **Traffic Duration**（流量时长）控制流量断续流的时长，而属性 **Traffic Mix**（混合流量）指定如何仿真这个流量断续流（见 6.6.1 节）。属性 **Traffic Start Time**（流量开始时间）指定这个流量断续流将开始执行的时间。它有几个预设值，包括如下几个：

1）Never（从不）指明这个需求将不产生任何流量。

2）**Same As Global Setting**（与全局设置相同）指明该流量断续流将在通过如图 6.23 所示的一个全局输入属性 **Background Traffic Start Delay**（seconds）[背景流量开始延迟（秒）] 指定的时间开始。在开始仿真之后的这么多秒时，流量断续流将开始执行其指定的流量概要。属性 **Background Traffic Start Delay**（seconds）[背景流量开始延迟（秒）] 是通过 **Configure/Run DES** 窗口指定的整个仿真的一个全局输入属性。

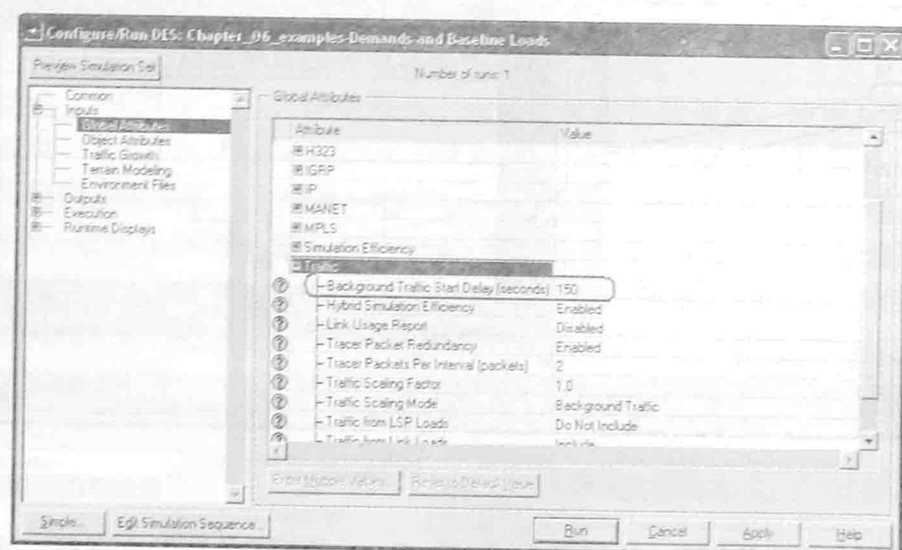


图 6.23　**Background Traffic Start Delay**（seconds）[背景流量开始延迟（秒）] 属性

6.6.3　基线负载

基线负载是建模一条链路的背景流量的一种机制。基线负载是在每条链路的基础上定义的，因此在一条链路上配置背景流量要求改变那条链路的属性值。如您所想到的，背景流量是不在网络中产生任何显式消息的条件下，使用一种纯分析性的方法对数据交换建模的。背景流量的影响被建模为穿越那条链路的每条显式产生的报文上的额外延迟。

每条链路模型包括称为 **Traffic Information**（流量信息）的一个属性，以基线负载的形式配置背景流量。可创建这个属性的多个实例（通过创建多个行），来为同一链路定义多个基线负载。每个基线负载是通过如下子属性定义的：

1）**Traffic Class**（流量类）指定这个基线负载的流量类型。这个信息由仿真引擎使用，对背景流量的 QoS 处理进行建模。

2）**Average Packet Size**（**bytes**）[平均报文尺寸（字节）] 定义背景流量中一条报文的平均尺寸。默认情况下，平均报文尺寸是 576 字节。

3）**Traffic Load**（**bps**）[流量负载（bps）] 以比特/秒为单位指定背景流量的速率。

子属性 **Average Packet Size**（**bytes**）和 **Traffic Load**（**bps**）是针对链路上每个方向的流量断续流独立指定的。例如，图 6.24 给出了节点 *Server*（服务器）和 *The Internet*（互联网）之间链路之基线负载的配置。这条链路称作 *Server* < - > *The Internet*。配置这条链路上基线负载的行由两个子属性组成：定义从 *Server* 到 *The Internet* 的流量的一个子属性（称作 *Logical Network. Server* - > *Logical Network. The Internet*），以及从 *The Internet* 到 *Server* 的流量的另一个子属性（称作 *Logical Network. The Internet* - > *Logical Network. Server*）。这两者具有 **Average Packet Size**（**bytes**）和 **Traffic Load**（**bps**）子属性的独立实例。

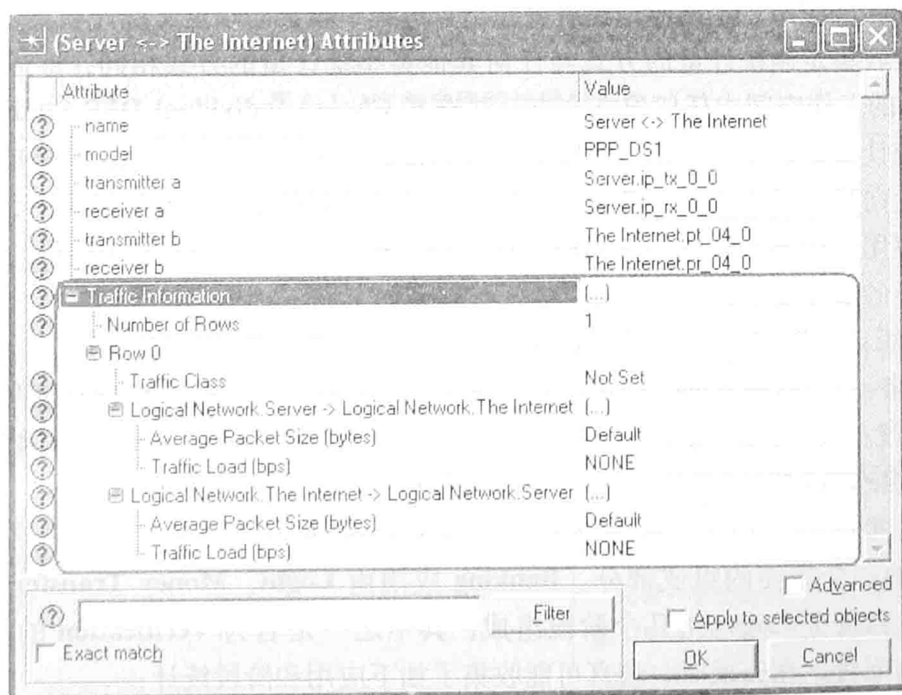


图 6.24　配置基线负载的属性

Traffic Load（**bps**）子属性的值也称作流量密度，是通过 **Traffic Intensity Attribute Profile**（流量密度属性概要）窗口定义的，该窗口定义流量密度如何随时间变化。**Traffic Intensity Attribute Profile** 窗口与需求流量概要的窗口（见图 6.22）相同。总

之，基线负载流量密度属性以需求流量概要相同的方式进行配置。OPNET称这种类型的流量负载为“静态吞吐量”，原因是它不对显式流量进行建模，相反仅简单地作为链路的占用有多繁重的一个指示。不像流量需求的是，基线负载仅以bit/s定义流量速率，而不是也使用报文数/s。这主要是因为基线负载的配置包括报文尺寸，在整个仿真期间保持不变，因此与以bit/s的流量速率一起，可被用来计算以报文数/s为单位的流量速率：

$$\text{流量速率（报文数/s）} = \frac{\text{流量速率（bit/s）}}{\text{报文尺寸（bit/报文）}}$$

一般而言，配置链路基线负载的过程是十分直接的，它由如下步骤组成：

1）在关注的链路对象上右击，并选择 **Edit Attributes** 选项。

2）展开属性 **Traffic Information**（流量信息）。回顾一下，在属性的值字段上双击会打开以表形式给出的相应子属性的一个窗口。

3）设置属性 **Number of Rows** 值，它代表要在这条链路上定义的基线负载总量。

4）为定义基线负载，一次展开一行，并为每行指定子属性值。

5）单击 **OK** 按钮保存改变，并关闭 **Edit Attributes** 窗口。

6.7 定制应用统计项

类似于标准应用，定制应用也有要在仿真过程中收集的预定义统计项集合。一个仿真场景配置收集定制统计量的方式与任何其他类型统计量的方式相同，即通过在 **Project Editor** 的工作空间中任何地方右击，并选择 **Choose Individual DES Statistics** 选项。被收集的统计量经常基于应用、阶段或任务名进行归组。为每项应用、任务或概要收集独立的统计向量。相应的识别符名字被附接在统计量名的末尾。典型情况下，依据如下惯例定义应用、任务和阶段的识别符，其中每个相应的名字被替换为概要、定制应用、任务或阶段的实际名字：

1）应用：<*Profile Name/Application Name*>（<概要名/应用名>）

2）任务：<*Profile Name/Application Name/Task Name*>（<概要名/应用名/任务名>）

3）阶段：<*Profile Name/Application Name/Task Name/Phase Name*>（<概要名/应用名/任务名/阶段名>）

例如，考虑一项仿真，配置运行名为 **Banking** 的一项定制应用，是名为 **Online Customer** 的一个概要的组成部分。**Banking** 应用由 **Login**、**Money Transfer** 和 **Logout** 任务组成，且任务 **Login** 由几个阶段组成，其中之一是名为 **Verification** 的一个阶段。基于上述的配置，在完成时，仿真可能收集了如下应用和阶段统计：

1）**Application Response Time**（**Seconds**）<**Online Customer/Banking**>（应用响应时间（秒）<在线客户/理财>），收集在概要 **Online Customer** 内定义的 **Banking** 应用的响应时间。

2）**Phase Response Time**（**seconds**）<**Online Customer/Banking/Login/Verification**>（阶段响应时间（秒）<在线客户/理财/登录/验证>），收集 **Login** 任务 **Verification**

阶段的响应时间，该任务是 **Online Customer** 概要内定义的 **Banking** 应用的组成部分。

一般而言，OPNET 提供如下所列四种不同类型的定制应用统计量：

1）称为 *Custom Application* 的全局统计量，包含部署于被仿真网络中定制应用所有实例的结果。取决于统计类型，这些结果是在从参与特定定制应用的数据传递中所有节点收集数据之上进行聚集或平均得到的。

2）称为 *Custom Application* 的节点统计量，包含有关部署于一个特定节点处定制应用的结果。

3）称为 *Requesting Custom Application* 的节点统计量，包含有关如下情形中的仿真结果，这些情形中当前节点作为定制应用各阶段的一个源。具体而言，当访问当前节点时，仅有源自这个节点的那些阶段的统计量才可进行查看。为了查看一个定制应用所有阶段的这些统计量，要求在所关注定制应用内作为一个阶段的源的每个节点处检查所收集的结果。

4）称为 *Responding Custom Application* 的节点统计量，包含与如下情形有关的仿真结果，在这些情形中当前节点作为定制应用阶段的一个目的地。具体而言，当访问当前节点时，仅有当前节点作为其目的地的那些阶段的统计量才可进行查看。为了查看一个定制应用所有阶段的这些统计量，要求在所关注定制应用内作为一个阶段的目的地的每个节点处检查所收集的结果。

表 6.1 给出所有可用定制应用统计量的汇总。

表 6.1 定制应用统计量的汇总

分类	名字	描述
Global Statistics Custom Application（全局统计量定制应用）	*Application Response Time* (*seconds*)［应用响应时间（秒）］	完成一个定制应用内所有任务执行所花费的时间。这个统计量是在网络中活跃的定制应用所有实例之上聚集得到。如果在所有定制应用任务之前仿真结束，那么这个统计量将报告任何值
	Packet Network Delay (*seconds*)［报文网络延迟（秒）］	这个定制应用的请求和响应报文所经历的网络延迟
	Phase Response Time (*seconds*)［阶段响应时间（秒）］	完成一项任务内一个阶段花费的时间。基于 **End Phase When** 属性的值，记录这个统计项的值。在某些情形中［如当 Final Request Leaves Source（最后的请求离开源）时完成阶段］，则一个阶段结束而它的一些消息仍然在网络中传输，就是可能的。这个统计量是在每个阶段基础上报告的

（续）

分类	名字	描述
Global Statistics Custom Application（全局统计量定制应用）	*Task Response Time* (*seconds*)［任务响应时间（秒）］	完成一个定制应用的单个任务内所有阶段所花费的时间。这个值是网络中在支持这个定制应用的所有节点上平均得到的。这个统计量是在每个任务基础上报告的
	Traffic Received (*bytes/sec*)［接收到的流量（字节/秒）］ *Traffic Received* (*packets/sec*)［接收到的流量（报文数/秒）］ *Traffic Sent* (*bytes/sec*)［发送的流量（字节/秒）］ *Traffic Sent* (*packets/sec*)［发送的流量（报文数/秒）］	流量到达和离开速率，是在支持这个定制应用的所有节点上平均得到的
Node Statistics Custom Application（节点统计量 定制应用）	*Active Custom Application Instance Count*（活跃的定制应用实例计数）	在当前节点处这个定制应用之实例总数
	Application Response Time (*seconds*)［应用响应时间（秒）］	在部署于这个节点的定制应用内所有任务完成执行所花费的时间
	Phase Response Time (*seconds*)［阶段响应时间（秒）］	在一个任务内完成一个阶段所花费的时间。类似于相应的全局统计量，这个统计量也是基于 **End Phase When** 属性值记录的。这个统计量是为部署于这个节点处定制应用所有节点记录得到的
	Task Response Time (*seconds*)［任务响应时间（秒）］	在一个定制应用单个任务内完成所有阶段所花费的时间。这个统计量是在每个任务基础上报告得到的
Node Statistics Requesting Custom Application（节点统计量请求定制应用）	*Request Generation Time* (*seconds*)［请求产生时间（秒）］	产生单条消息花费的时间。如您所想到的，一个定制应用内一个阶段的每条请求消息由几条报文组成。这个统计量记录从请求消息的第一条报文被发出的时间开始到请求消息的最后一条报文的时间为止发送一条消息的总时间。这个统计值是仅为由这个节点产生的请求消息计算而不是为定制应用的所有请求消息计算的。这个统计量是在每个阶段基础上记录得到的

（续）

分类	名字	描述
Node Statistics Requesting Custom Application（节点统计量请求定制应用）	*Request Response Round Trip Time* (*seconds*)[请求响应往返时间（秒）]	一次请求—响应消息交换所花费的往返时间。这个统计值是如下计算的：从响应消息的最后一条报文到达的时间减去请求消息的第一条报文发送的时间。这个统计量是在每个阶段基础上记录得到的
	Request Size (*bytes*)[请求尺寸（字节）]	在一个定制应用阶段过程中产生的一条请求消息的尺寸。这个统计量是这样计算的：将属于一条特定请求消息的所有报文的尺寸求和。这个统计量是在每个阶段基础上记录得到的
	Response Packet Network Delay (*seconds*)[响应报文网络延迟（秒）]	作为源自这个节点的请求的应答，到达这个节点的响应报文所经历的网络延迟。这个统计量也是在每个阶段基础上记录得到的。如果一个阶段被配置为不产生响应消息，那么这个统计量将不记录任何值
	Traffic Received (*bytes/sec*)[接收到的流量（字节/秒）] *Traffic Received* (*packets/sec*)[接收的流量（报文数/秒）] *Traffic Sent* (*bytes/sec*)[发送的流量（字节/秒）] *Traffic Sent* (*packets/sec*)[发送的流量（报文数/秒）]	流量到达和离开速率。仅对在这个节点处产生的流量或到达这个节点的流量计算这些统计量。这些统计量是在每个应用基础上收集的，即在当前节点处所部署应用接收/发送的流量
Node Statistics Responding Custom Application（节点统计量响应定制应用）	*Load* (*request/sec*)[负载（请求数/秒）]	在这个节点处请求消息的到达率。这个统计量是在每个应用基础上记录的
	Load (*sessions/sec*)[负载（会话数/秒）]	在这个节点处所创建新阶段的速率。典型情况下，当一个做出响应的节点从一个阶段的第一条请求消息接收到一条报文时，在该节点处开始的一个新阶段。因此，这个统计量收集在这个节点处第一条请求消息的到达率。这与 **Load（requests/sec）** 统计量是不同的，后者收集 ALL（所有）请求消息的到达率。这个统计量是在每个应用基础上记录的

（续）

分类	名字	描述
Node Statistics Responding Custom Application（节点统计量响应定制应用）	*Request Packet Network Delay* (*seconds*)［请求报文网络延迟（秒）］	到达这个节点的请求报文所经历的网络延迟。这个统计量也是在每个阶段基础上记录得到的
	Request Processing Time (*seconds*)［请求处理时间（秒）］	处理一条请求消息花费的时间。这个统计量是如下计算的：从响应响应消息的最后一条报文从这个节点发出的时间，减去请求消息的最后一条报文到达该节点的时间。这个统计量也是在每个应用基础上记录得到的
	Response Size (*bytes*)［响应尺寸（字节）］	在一个定制应用阶段过程中产生的一条响应消息的尺寸。这个统计量是如下计算的：将属于一个特定响应消息的所有报文的尺寸求和。这个统计量是在每个阶段基础上记录得到的
	Traffic Received (*bytes/sec*)［接收到的流量（字节/秒）］ *Traffic Received* (*packets/sec*)［接收的流量（报文数/秒）］ *Traffic Sent* (*bytes/sec*)［发送的流量（字节/秒）］ *Traffic Sent* (*packets/sec*)［发送的流量（报文数/秒）］	流量到达和离开速率。仅对在这个节点处产生的流量或到达这个节点的流量计算这些统计量。这些统计量是在每个应用基础上收集的，即在当前节点处所部署应用接收/发送的流量

6.8　应用和流量需求的统计量

OPNET 也提供应用和流量需求的统计量。应用需求统计量可作为通用节点统计项的组成部分，而又称为 *Demand Statistics*（需求统计量）的一个独立类，具有流量需求的统计信息。如果在仿真中没有部署流量需求，那么需求统计类在 **Choose Results** 窗口中就是不可用的。在仿真执行之后，如果进行了配置，则通过检查作为应用需求的源或目的地的各节点的结果，可查看应用需求统计信息。另外，流量需求统计量被显示为对象统计项之下的一个独立类，它们依据如下惯例进行命名：< *Name of the Source Node* > - > < *Name of the Destination Node* >（< 源节点名 > - > < 目的地节点名 >）。表 6.2 汇总了可用的需求统计量。

表 6.2　应用和流量需求统计量的汇总

分类	名字	描述
Node Statistics Application Demand（节点统计量应用需求）	*Response Time*（*seconds*）[响应时间（秒）]	在一条请求被发送之后一条响应消息到达的时间。这个统计量是如下计算的：从源节点接收一条响应消息（是其请求的应答）的时间，减去源节点产生请求的时间
	Traffic Received（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）]	一个应用请求的流量到达速率
	Traffic Sent（*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	一个应用请求的流量离开速率
Demand Statistics（需求统计量）	*Packet End - to - End Delay*（*sec*）[报文端到端延迟（秒）]	在源节点产生一条报文与报文到达目的地节点之间的时间。这个统计量仅对非背景流量是可用的，即如果流量需求被配置为 All Background（都是背景）流量，那么将不记录这个统计量的值
	Packet Jitter（*sec*）[报文抖动（秒）]	到达目的地的两条连续报文的端到端延迟之差的绝对值。这个统计量也仅对非背景流量有效
	Traffic Received（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）]	一个流量需求的流量到达速率
	Traffic Sent（*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	一个流量需求的流量离开速率

6.9 显式报文产生源和基线负载的统计量

显式报文产生源和基线负载没有统计量。因此，为了研究这些流量源对网络性能的影响，需要检查其他统计量。IP 层、媒介访问控制（MAC）层和链路统计量可提供显式报文产生源所产生流量性能的有关反馈。基线负载对背景流量进行建模，且不产生任何显式报文。因此，IP 和 MAC 层统计量将不会提供有关基线负载流量的任何信息。但是，诸如排队延迟、吞吐量和利用率等的 *point - to - point*（点到点）链路统计量将显示所配置的基线负载如何影响在相应链路上传输的流量。

这条规则的唯一例外是 *ethcoax_ station* 模型，该模型仿真 ON - OFF 显式报文产生源。将这个模型添加到仿真，使两个新的全局统计和节点统计类可用于收集 **Traffic Sink**（流量目的）和 **Traffic Source**（流量源）。这些新的附加统计量支持收集这样的信息，如端到端延迟以及到达和传输流量速率。这些统计量的含义是自解释的，基于上述所提供的类似统计量的描述，应该能够区分它们的目的。

第 7 章　指定用户概要和部署应用

7.1　用户概要

一个 **user profile**（用户概要）是指定在一个仿真过程中标准应用和定制应用如何为一名端用户所用的一种机制。虽然有时简单地称它们为概要，用户概要不应该与其他类型的概要（如流量概要）相混淆。回顾一下，配置标准应用和定制应用的过程由如下步骤组成：①定义应用；②配置在仿真过程中要收集的应用统计量；③定义用户概要；④部署所定义的应用（实际上涉及部署用户概要）。第 5 章和第 6 章讨论配置标准应用和定制应用的过程。本章焦点为定义用户概要和在被仿真网络系统内部署应用。

应用定义指定所产生流量的特征，而用户概要定义这些应用如何由终端用户执行。这包括指定诸如应用的开始时间和时长、单个用户执行的应用数量、这些应用被执行的顺序等的信息。例如，一项电子邮件应用的定义也许指定了电子邮件客户端每隔 10s 接收尺寸为 1024B 的 5 条电子邮件消息，这产生 0.5KB 的流量到达率。但是，电子邮件应用的定义不会指定在被仿真网络中有多少用户运行这项电子邮件应用，每名用户何时开始和结束执行这项应用，用户重新运行该应用有多少次，是否任意用户都可运行多项应用，单个用户执行多项应用是并行的还是串行的，等等。这种信息是通过用户概要和应用部署过程指定下来的。

7.2　指定用户概要

正常情况下，仅在仿真内采用的所有应用都被定义之后，才配置用户概要，原因是用户概要的某些属性要被设置为所定义应用的名字。但是，如果需要，一个已有的用户概要可被重新配置以便包括新的应用定义。在本节，我们描述如何指定和配置用户概要。

7.2.1　Profile Config 工具对象

类似于应用定义，配置用户概要也要求存在一个工具对象。图 7.1 给出一个 *Profile Config*（概要配置）对象的图标，支持配置用户概要。典型情况下，*Profile Config* 对象位于 **Object Palette Tree** 的与 *Application Config* 对象一样的文件夹中（见 5.3.1 节）。类似地，为了配置所有必要的用户应用，每个仿真场景仅需要一个 *Profile Config* 对象。



图 7.1　*Profile Config* 模型图标

通过指定 *Profile Config* 对象的属性值，定义用户概要。以与任何其他节点的属性访问方式相同的方式，访问 *Profile Config* 对象的属性：右击 *Profile Config* 对象，并选择 **Edit Attributes** 选项。图 7.2 给出 *Profile Config* 对象的属性。

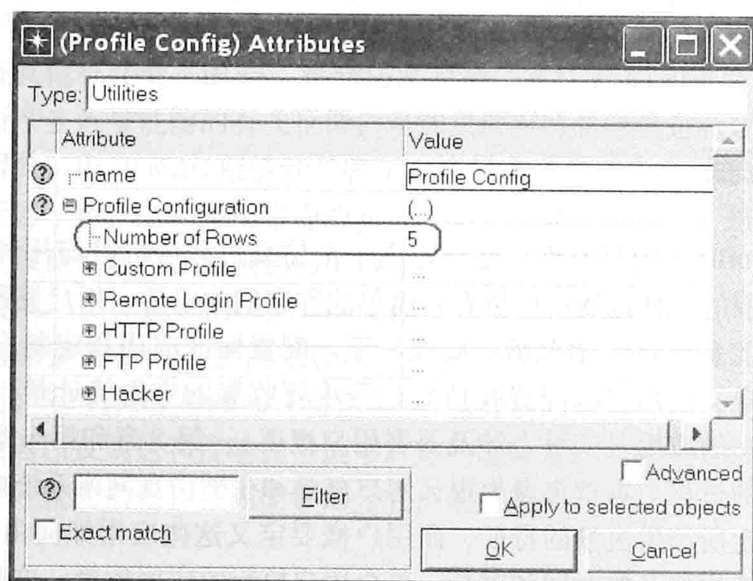


图 7.2 *Profile Config* 对象属性

如图 7.2 所示，Profile Config 对象包含两个属性：**name**（名）和 **Profile Configuration**（概要配置）。第一个属性指定对象出现在项目工作空间中时的名字，第二个属性包含为当前仿真场景配置的用户概要的定义。复合属性 **Profile Configuration** 的 **Number of Rows**（行数）子属性控制为这项仿真研究可配置的用户概要数量。在图 7.2 所示的例子中，**Number of Rows** 属性被设置为 5，对应于在 *Profile Config* 对象内作为独立的复合属性定义的 5 个用户概要，即 Custom Profile（定制概要）、Remote Login Profile（远程登录概要）、HTTP Profile（HTTP 概要）、FTP Profile（FTP 概要）和 Hacker（黑客）。

7.2.2 定义一个用户概要

一个用户概要是这样定义的，赋予它一个名字，指定要包括在概要中的应用以及描述概要的整体行为。仔细研究图 7.2 中概要定义之一，以便更好地理解用户概要的配置。图 7.3 给出 Remote Login Profile 的展开视图。

如图 7.3 所示，一个概要定义中的第一个属性是 **Profile Name**（概要名字），它对应于整个概要定义行的名字。默认情况下，每个新的概要行得到名字“Enter Profile Name...”（输入概要名...），但设置属性 Profile Name 的值，也会使概要定义的名字发生改变。

在一个概要定义中的下一属性被称为 **Applications**（应用）。这是指定应用的一个复合属性，是概要的组成部分；它也描述那些应用中每项应用的行为，以及在概要中各

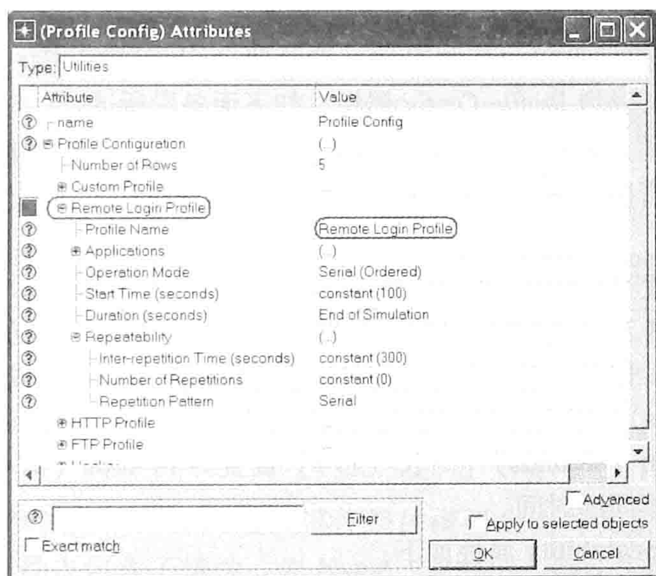


图 7.3　Remote Login Profile 的定义

应用如何被使用。常见的情况是，单个概要执行多项应用，因为在真实生活中，用户频繁地在单个计算机会话过程中运行多项应用。Applications 属性的详细描述以及如何用之指定应用行为在 7.2.4 节和 7.2.5 节给出。

最后，概要定义包含了一个属性集合，用来指定概要的整体行为。这些属性并不描述在概要中的个体应用的行为如何（是通过复合属性 **Applications** 指定的），但相反它们指定在仿真过程中如何管理整个概要。概要行为属性如下：

1）**Operation Mode**（工作模式）属性控制应用在概要内执行的方式。一些这样的应用是顺序执行的，例如在上午，一名办公室工作人员首先通过 Web 阅读公司通告，之后登录到远程服务器进行日常工作，而其他应用可能是并行运行的，例如，在操作数据库应用时，该名工作人员也运行电子邮件应用，周期性地检查重要的消息。如果概要仅包含一项应用，那么 Operation Mode 的值就是无关的。

2）**Start Time**（**seconds**）[开始时间（秒）] 指定相对于仿真开始时间的用户概要将开始的时间。

3）**Duration**（**seconds**）[时长（秒）] 指定概要将运行多长时间。

4）**Repeatability**（可重复性）是一个复合属性，描述如果用户概要在仿真结束之前完成，该概要将如何重复。

在 7.2.3 节说明上述属性如何用来定义一个简单的用户概要。在 7.2.6 节给出概要行为属性的一个比较详细的描述。

7.2.3　一个简单用户概要例子

考虑如下一个简单用户概要的例子：假定一名办公室秘书 Lisa，正常情况下在其工作日仅允许两个应用——网页浏览和电子邮件。Lisa 在大约上午 8:00 开始在办公室工

作，在大约下午 5:00 下班回家。当她到达办公室时，假定在她能够做任何工作之前需要花费 2min 启动她的计算机。在此之后，她分两个 4h 时段工作，中间有 1h 的休息。这种行为可容易地映射到 *Profile Config* 属性，如下面各段所述。

第一步是命名用户概要。出于可读性考虑，命名这个概要为 Lisa。接下来，配置 **Application** 属性运行网页浏览和电子邮件。假定相应的应用定义已经在 *Application Config* 对象中创建。

在此之后，设置 **Operation Mode** 为 Simultaneous（同时的），表明 Lisa 同时运行这些应用。Lisa 的典型工作日是 9h，包括 1h 的午饭时间。但是，在仿真中，也需要 2min 启动 Lisa 的计算机。因此，仿真的总长度应该被设置为 9h 2min 或 542min。将概要的 **Start Time**（**seconds**）［启动时间（秒）］设置为 120s，代表花费 2min 启动计算机和应用。概要的 **Duration**（**seconds**）［时长（秒）］设置为 14 400s（即 4h），这是在午饭前后 Lisa 用于工作的不间断时间。

最后，配置 **Repeatability** 属性如下：

1）**Inter – repetition Time**（**seconds**）［重复间隔时间（秒）］被设置为 3600s（即 1 小时），指明 1 小时的午饭休息时间。

2）**Number of Repetitions**（重复次数）被设置为 1，指明 Lisa 将工作 1 次以上的不间断时段（即从大约下午 1:00 到下午 5:00）。

3）**Repetition Pattern**（重复模式）被设置为 serial（串行），指明概要实例一次次地重复。

图 7.4 给出这个概要配置的汇总情况，图 7.5 给出概要执行的一条时间线。注意，如果在概要结束之前仿真时间用完，则在不等待概要结束的条件下，仿真将终止。

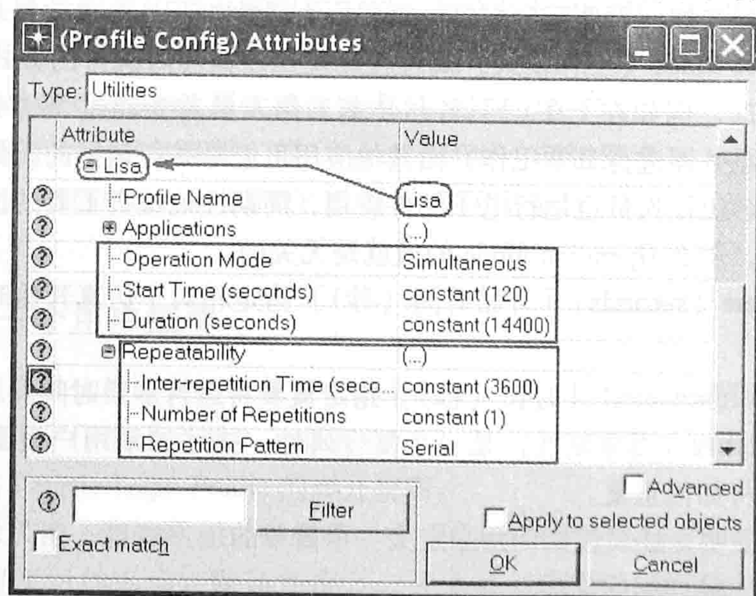


图 7.4 一个概要配置例子

7.2.4 在一个概要内配置应用行为

既然有了仿真和概要时间线之间关系的基本理解，那么将另一个技巧（wrinkle）

添加到这个配置过程：在概要内指定应用行为。在一个概要定义中 **Application** 复合属性包含一个子属性 **Number of Rows**（它控制在当前概要内包含的应用数量）和一个行集合，每行描述单个应用将如何用在这个概要中。应用描述行的数量对应于子属性 **Number of Rows** 的值。

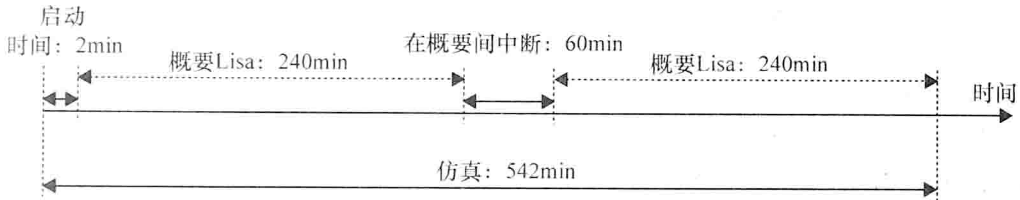


图 7.5　图 7.4 中概要的执行时间线

首先，尝试理解 **Application** 属性及其环绕的概要之间的关系。类似于概要定义，**Applications** 复合属性也包含子属性，如 **Name**（名字）、**Start Time Offset**（**seconds**）[开始时间偏移（秒）]、**Duration**（**seconds**）[时长（秒）] 和 **Repeatability**（可重复性），如图 7.6 所示。主要区别是，在概要定义内，这些属性描述相对于仿真开始时间的概要的行为，而在 **Applications** 的情形中，这些属性描述相对于当前概要的开始时间的应用行为。因此，应用的属性 **Start Time Offset** 指定在当前概要启动之后何时应用将开始执行的时间。例如，考虑这样一种情况，其中概要的 **Start Time**（**seconds**）被设置为 120，而其应用的 **Start Time Offset**（**seconds**）设置为 300。在这种情况中，概要和应用将分别在仿真开始之后的 120s 和 420s 开始执行。类似地，Repeatability 描述在应用的概要内应用的重复模式。

在 7.2.5 节中比较详细地描述上述应用行为属性。

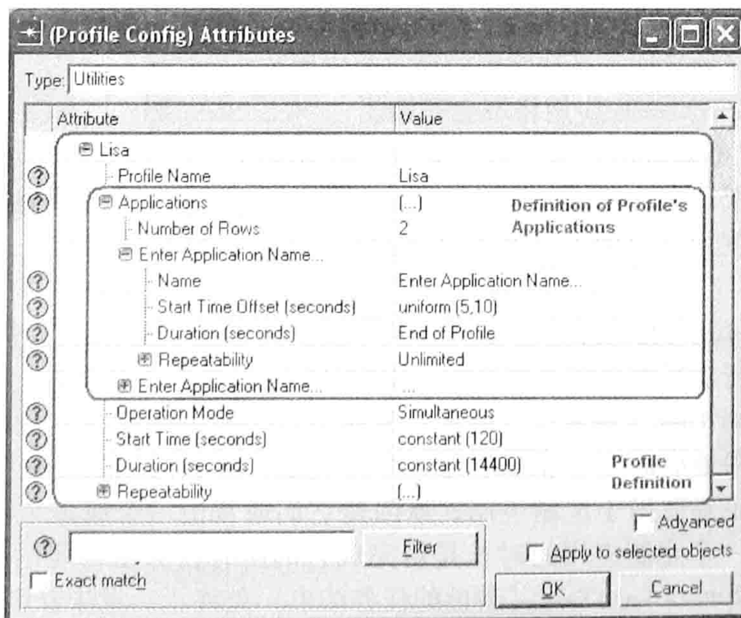


图 7.6　在概要内 **Applications** 属性的定义

7.2.5 应用行为属性

描述概要中每个应用行为的属性是复合属性 **Applications** 的组成部分。Applications 属性的第一个子属性是 **Number of Rows**，指定要在这个概要内配置的应用数量。为每个应用创建一个独立的复合属性。这个属性最初称为 **Enter Application Name...**（输入应用名字...），它包含概要内一个应用的完整用法描述。

一个应用定义的各组成如下：

1）**Name**——在这个概要内要使用的应用的名字。这个属性包含已经通过 *Application Config* 对象配置的各应用的一个列表。只能从所提供的列表中选择一个应用。如果不存在期望的应用，那么需要在 *Application Config* 对象中定义它，只有这时才可返回配置用户概要。一旦选择了期望的应用，则应用的名字将从 **Enter Application Name...** 改变为所选择的应用名。

2）**Start Time Offset**（**seconds**）［开始时间偏移（秒）］属性的含义取决于 **Operation Mode** 属性的值（见7.2.6节），后者指定各应用在概要内如何执行：同时（即并行）或串行（即一个接着另一个）。如果应用执行是同时的，那么 **Start Time Offset**（**seconds**）属性指定从概要的开始时间到这个应用开始执行之前的时段长度。如果应用是串行执行的，那么这个属性指定从当前概要内前一应用的结束到这个应用开始执行之前的时段长度。如果这是列表中的第一个应用，那么这个应用将在概要开始之后 **Start Time Offset**（**seconds**）开始执行。如果在概要结束之前前一应用没有终止，那么当前应用将不会启动。更具体地说，如果在前一应用完成的时间与 **Start Time Offset**（**seconds**）属性的值之和超过当前概要完成的时间，那么这个应用将不会执行。同样，这个属性有两个预定义值：

① No Offset（没有偏移）——应用将与概要同时开始（如果在概要内的各应用是同时执行的，或如果这是列表中的第一个应用），或一旦前一应用完成，则开始（如果各应用执行是串行的且这不是列表中的第一个应用）。

② Never（从不）——应用将不会开始。

3）**Duration**（**seconds**）指定这个应用的单个实例将执行的时长。注意在概要内一个应用可重复多次，在后来的时间可创建应用本身的新实例。这个属性可被设置为一个显式的属性值［使用一个确定的概率分布函数（PDF）计算得到的］或它可从两个预设选项中选择：

① End of Profile（概要结束）意味着这个应用将在这个概要的同时终止。如果在概要内的应用是串行执行的，那么在当前应用将没有定义的应用会执行。

② End of Last Task（最后任务结束）意味着这个应用将仅在其最后的任务完成后终止。但是，如果最后的任务在概要结束后才完成，那么应用将在概要终止的同时终止。这个属性值主要适用于定制应用，原因是它们通常由一个或多个任务（即节点间的事务）组成。一个定制应用的时长是由完成它的所有任务所需要的时间控制的。因此，设置应用在 End of Last Task（最后的任务结束）处终止，意味着您希望在它结束它的所有网络事务或活动之后终止应用。对于没有显式定义的任务的标准应用，End of Last Task 设置与 End of Profile 具有相同含义。

4）**Repeatability** 是一个复合属性，描述当前应用如何在所环绕的概要内重复（即在概要正在被执行时）。具体而言，OPNET 通过如下三个子属性定义 **Repeatability**：

① **Inter - repetition Time**（**seconds**）［重复间隔时间（秒）］属性指定应用会话间时段的时长。这个属性是通过一个 PDF 指定的，在仿真过程中使用 PDF 计算重复间隔时间。这个属性的含义取决于重复模式，为 Serial（串行）或 Concurrent（并行）。如果这个应用是串行重复的（即仅在前一会话结束之后，下一应用会话才开始），那么 **Inter - repetition Time**（**seconds**）指定从前一应用会话完成直到下一会话开始的时段长度。另外，如果应用是并行重复的（即多个应用会话可以是同时处于活跃状态的），那么 **Inter - repetition Time**（**seconds**）指定从前一会话开始（*start*）到下一会话开始的时段长度。当应用是并行重复时，将 **Inter - repetition Time**（**seconds**）设置为 0，会导致以一个无穷大的速率创建应用会话，这可能导致人们不期望的仿真输出。

② **Number of Repetitions**（重复次数）指定这个应用将被重复的次数。重复次数不同于在概要生命期间创建的应用会话数。重复次数仅统计在第一个应用会话开始之后的会话。例如，如果重复次数设置为 1，那么在概要生命期间，至多将执行两个应用会话（即如果第一个会话在概要结束之前没有完成且应用使用串行重复模式执行，则可能仅有一个会话将执行）。这个属性是通过一个 PDF 指定的，但也可有称为 Unlimited（无限的）的一个预定义值，这意味着直到当前用户概要结束之前，该应用将继续重复执行。

③ **Repetition Pattern**（重复模式）指定在这个概要内当前应用将如何重复。这个属性仅有两个可能值：Serial（串行）（指明仅在前一应用完成之后，下一应用才可开始）和 Concurrent（并行）（意味着多个应用会话可同时执行，且下一应用可在前一应用完成之前开始）。图 7.7 所示为并行和串行应用重复模式之间的差异。

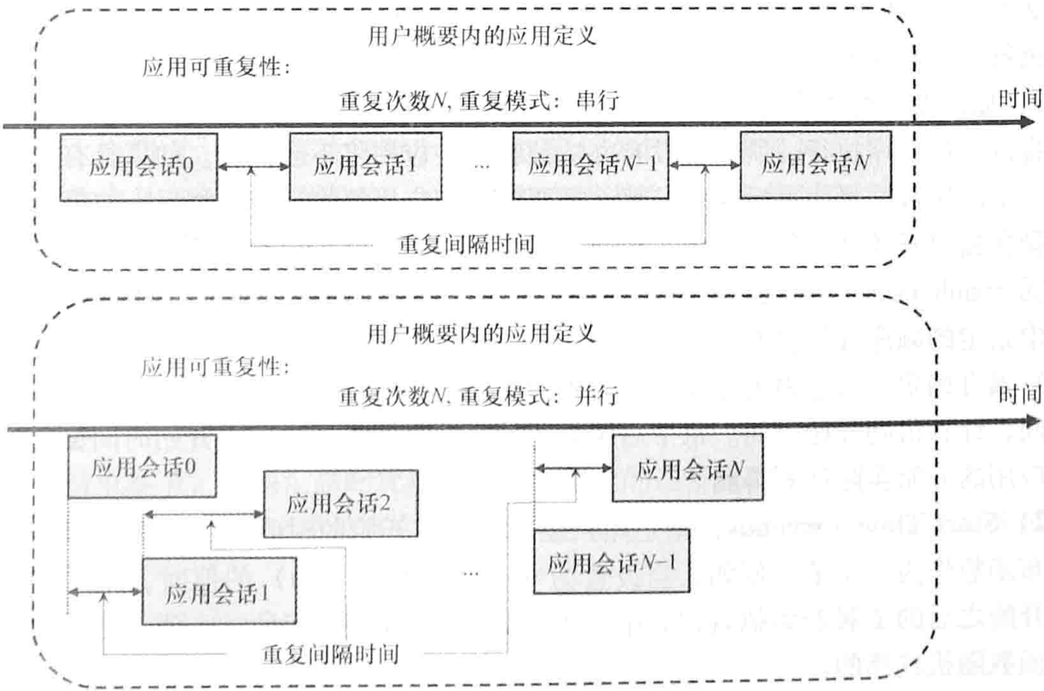


图 7.7　在概要内的应用可重复性

最后，复合属性 **Repeatability** 具有两个预设值：

① Once at Start Time（在开始时间重复一次）意味着在整个概要时长上应用将仅执行一次。通过将子属性 **Number of Repetitions** 设置为 0，可容易地取得相同效果，事实上是这个值是如何设置的。

② Unlimited（无限制的）意味着直到概要结束之前该应用将继续重复执行。配置这个预设置，是使应用以一种串行顺序重复，重复间隔时间是使用均值输出为 300s 的一个指数分布计算得到的。实际上，当选择这个值时，直到概要结束之前，将在前一会话完成之后以平均 300s 开始下一应用会话的方式重复应用。

7.2.6 概要行为属性

如在 7.2.2 节中简要介绍的，概要行为属性是一个概要定义的组成部分，并被用于描述一个概要的整体行为。现在比较详细地描述这些属性：

1）**Operation Mode**（工作模式）属性描述所定义的概要应用将如何在概要内开始。这个属性没有描述任意一个个体应用的行为，相反，它配置概要内所有应用的执行模式。这个属性有三个可能值：

① Serial（Ordered）［串行的（有序的）］——各应用将以在 **Applications** 属性中指定的顺序一个接一个地执行，即在第一行中定义的应用首先执行，接下来是在第二行中定义的应用，等等。如果一个应用被配置为重复无限次或其时长被设置为 End of Profile（概要结束），那么后续应用没有哪个将开始，原因是前面的应用从来就不会在概要生命期内完成。

② Serial（Random）［串行的（随机的）］——各应用将以一个随机顺序一个接一个地执行。如果至少有一个应用被配置为无限次的可重复性或其时长被设置为 End of Profile，那么在概要内指定的一个或多个应用都不会执行的危险是仍然存在的。因为各应用将以随机顺序执行，所以就没有办法预料哪些应用将不会执行。如果具有无限次重复的一个应用被选择为最后执行，那么概要的所有应用都将运行，相反，如果那个应用被选择在概要开始处执行，那么它将是在那个概要期间运行的唯一应用。

③ Simultaneous（同时的）——各应用将同时执行。各应用将以它们在 **Applications** 属性中指定的顺序开始执行。每个应用的开始将由那个应用的 **Start Time Offset（seconds）**属性确定。对于概要中的第一个应用，通过将其开始偏移值加到当前概要发起的时间，计算得到开始时间。每个后续应用的开始时间是通过将其开始时间偏移值加到前一应用的开始实际计算得到。

2）**Start Time（seconds）**指定用户概要会话将开始的时间。这个属性接受一个概率分布函数作为一个值。例如，当设置为 uniform（100，110）的值时，用户概要将在仿真开始之后的 *T* 秒开始执行，其中 *T* 是在 100s 和 110s 之间的一个值，使用均匀概率分布函数随机选择的。

3）**Duration（seconds）**指定概要将执行多长时间。这个值覆盖应用时长，意味着即使在概要结束之前应用没有完成，概要和应用将依据概要的 **Duration** 属性中指定的

值而终止。这个值具有两个预定义值：

① End of Simulation——概要将继续执行，直到仿真结束。采用这样一个设置，所有应用将在概要结束之前完成并不会产生任何更多的流量，将是可能的；但是，直到仿真结束之前，概要将保持活跃状态。如果概要被配置为使用串行重复模式重复执行，建议不要将概要时长设置为 End of Simulation，原因是由于概要的第一个实例在仿真终止之前不会结束，所以后续的重复将不会执行。

② End of Last Application（最后的应用结束）——这个值指明，一旦其应用的最后一个实例结束，概要将终止。如果各应用被配置为重复无限次，那么概要将在仿真结束之前不会终止。但是，这是一个非常有用的预设值，因为它确保如果概要完成了它的所有工作且不再产生流量，那么概要将被禁止（deactivated）。

4）**Repeatability** 复合属性指定概要将在仿真内如何被重复。这个属性几乎等同于应用的 **Repeatability** 属性。唯一的区别是，这个属性描述 *the profile's repeatability within the simulation*（在仿真内概要的可重复性）而后一属性描述 *the application's repeatability within the current profile*（在当前概要内应用的可重复性）。这个属性也由三个子属性组成：

① **Inter – repetition Time（seconds）** 指定下一概要会话将开始的时间。如果概要被配置为串行重复的，那么下一概要会话的开始时间是如下计算得到的：将前一概要会话 *ended*（结束）的时间加到概要的重复间隔时间。如果各概要是并行执行的，那么下一概要会话的开始时间是如下计算得到的：将前一概要会话的 *started*（开始）时间加到概要的重复间隔时间。与前面一样，当各概要是并行重复时，将重复间隔时间设置为 0，将导致一个无穷大的概要产生率。

② **Number of Repetitions** 指定这个概要将被重复的次数。与应用的情况一样，重复次数仅统计在第一个概要会话开始之后的各会话。例如，如果重复次数被设置为 1，那么在仿真生命期内，至多将执行两个概要会话。这个属性将通过一个 PDF 指定，但也可有一个称为 Unlimited 的预设值，这意味着直到当前仿真结束之前，概要将继续重复。

③ **Repetition Pattern** 指定在这个仿真内当前概要将如何被重复执行。这个属性将仅有两个可能值：Serial（指明下一概要会话将仅在前一概要结束之后才开始）和 Concurrent（意味着多个概要将同时执行，且下一概要会话可在前一会话完成之前开始）。

7.2.7　配置用户概要

为了结束本节，下面汇总配置用户概要的完全的逐步骤的指令。

1）如果 *Profile Config* 还没有在项目工作空间中，那么将 *Profile Config* 对象放到项目工作空间。

2）右击 *Profile Config* 对象，并从弹出菜单中选择 **Edit Attributes**。

3）展开 **Profile Configuration** 复合属性。

4）指定 **Number of Rows** 属性的值，该值对应于要在当前仿真场景中配置的不同用户概要的总数。

5）一次展开一行，配置每个用户概要：

① 设置 **Name** 属性的值，这将自动地改变应用的行的名字。

② 展开复合属性 **Applications**，配置要在概要中使用的应用：

- 指定 **Number of Rows** 属性的值，该值对应于在这个概要中使用的应用总数。
- 对于每项应用，展开其相应的复合属性，并配置应用的每个属性。回顾一下属性 **Name**（指定应用的名字），仅能接受通过 *Application Config* 对象已经配置的应用名的预设值之一。同样，设置属性 **Name** 的值将自动地改变行的名为所提供的值。

③ 配置概要的 **Operation Mode**、**Start Time**（**seconds**）、**Duration**（**seconds**）和 **Repeatability** 属性。

6）直到所有要求的用户概要都被配置之前，重复上述步骤。

7.3　配置用户概要的例子

现在考虑配置用户概要的一个比较详细的例子。这个例子构造两个用户概要，对应于在一个工作地点的两名雇员的工作日程。

首先，考虑名为 Jon 的一名雇员的一个典型工作日。Jon 通过远程登录到其公司的计算机开始一天的工作。在工作大约 30min 之后，Jon 开始周期性（如每 30min）地查看网上的新闻。通常情况下，Jon 在每个这样的会话中浏览网页（Web）花费大约 10min。另外，每小时 Jon 都使用 FTP 应用备份其文件，这大约用去 2min。在浏览网页和传递备份文件时，Jon 保持远程登录应用处于活跃状态。Jon 通常以三班方式工作，每班 3h，每两班之间有大约 45min 的休息。假定在休息之前，Jon 关闭他的所有应用，之后在休息回来后重启这些应用。

这样一个用户概要可如图 7.8 所示配置。首先，详细研究捕获 Jon 的整体行为的概要执行模式：

1）因为在传递文件和浏览网页时，Jon 没有终止远程登录应用，所以所有应用都被配置为同时执行。

2）概要运行 10 800s，对应于 Jon 的 3h 班。

3）该概要串行地重复两次（即该概要总共将执行 3 次，顺次执行），重复执行之间有 2700s 的休息（即 45min）。

概要开始时间被设置为 100s 和 110s 之间的一个值，使用一个均匀分布计算得到。通常，将概要启动时间设置为一个非零值（即最好在 100s 左右），是一种不错的想法，这使被仿真系统中的其他设备（如路由器）进行初始化。注意概要本身仅描述单个 3h 班，而概要重复特征支持仿真一个完整的工作日。

现在，详细研究个体应用是如何在概要中配置的。因为 Jon 在每班时段过程中运行远程登录应用的单个实例，所以这个应用配置为在开始时仅开始一次，并在概要结束之前运行。注意，这样的一个配置不会防止其他应用开始执行，原因是概要被配置为同时执行应用。将远程登录的开始时间偏移设置为 10s，代表 Jon 用来开始应用的时间，虽

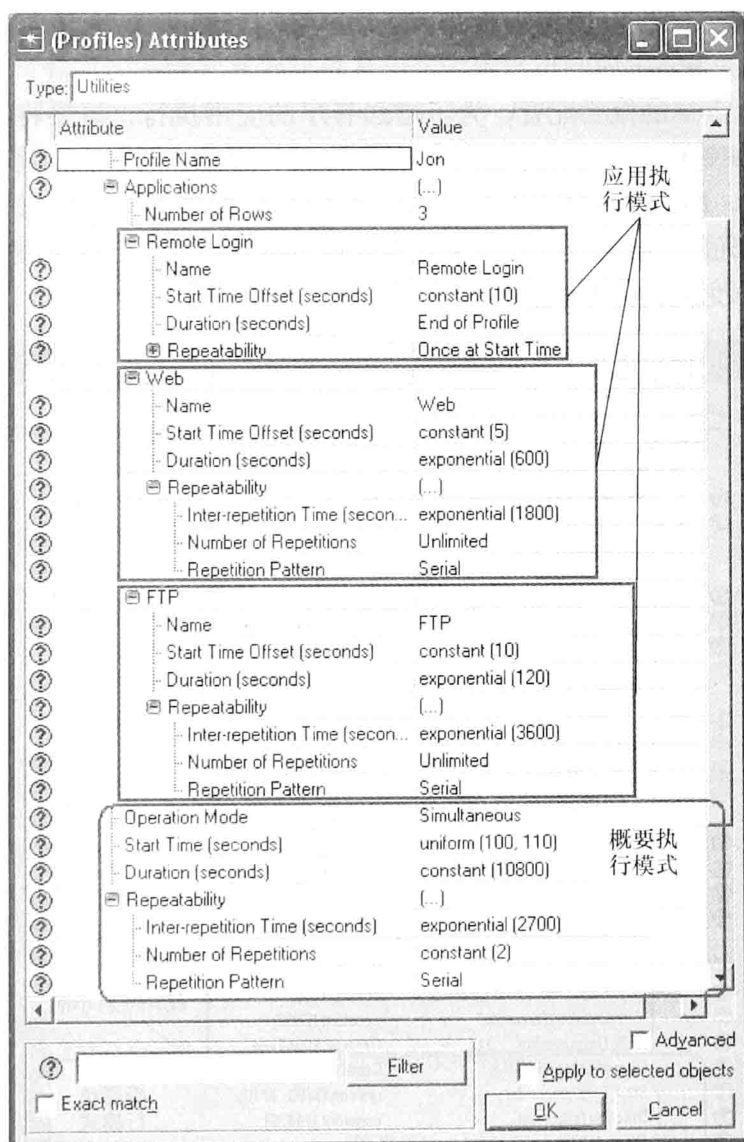


图 7.8　Jon 的用户概要的配置

然这个值已经被设置为 0。因为 Jon 每隔 30min 花 10min 浏览网页，所以配置 Web 应用有 600s（或 10min）的时长，并每 1800s（或 30min）重复无限次，这确保 Web 应用在概要结束（即一班时长的结束）之前将继续执行。因为 Jon 一次仅运行网页浏览器的一个实例，所以将重复模式设置为串行。注意 Web 应用的开始时间偏移被设置为 1800s。这仿真 Jon 的行为，其中他首先使用远程登录应用工作 30min，仅在此时开始浏览网页。使用同样的逻辑，配置 FTP 应用，该应用运行 120s（或 2min），并以串行方式重复执行，无限次地重复每个 3600s（或 60min）。FTP 开始时间偏移被设置为 3600s，指明 Jon 在 1h 的工作之后进行第一次文件传递。

现在考虑第二位雇员 Alice，她是在 Jon 之后 1h 开始工作的。Alice 以如下顺序一次运行一个应用：首先，她花 1.5h 远程连接到她公司的服务器，之后花费大约 10min 将文件传递到公司的服务器并从公司服务器传递文件到本地，接下来的 30min，她浏览网

页，之后再次花费 1h50min 使用远程登录应用。之后，她花费 1h 休息，后来在另一个 4h 时段重复相同过程。

图 7.9 给出 Alice 的概要配置。为了增强有序的应用执行，概要将其属性 Operation Mode 设置为 Serial（Ordered）[串行的（有序的）]，这意味着各应用将以它们被定义的顺序执行：*Remote Login*、*FTP*、*Web* 和 *Remote Login*。概要开始时间被设置为 3700s 和 3710s 之间的一个值，这仿真了 Alice 在比 Jon 晚 1h 开始工作，Jon 的概要开始时间被设置为 100s 和 110s 之间的一个值。

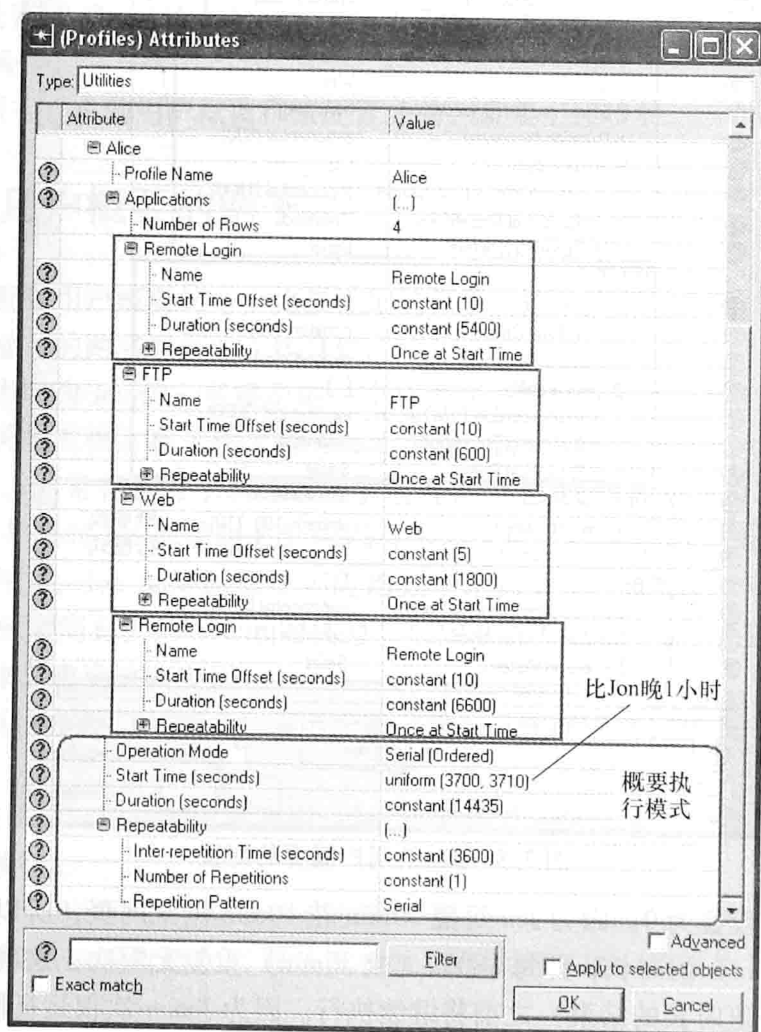


图 7.9　Alice 的用户概要配置

Alice 的概要时长被设置为 14 435s，这对应于 4h 和额外的 35s，这额外的 35s 统计的是应用开始之间的时间，即第一个 *Remote Login* 应用在概要开始之后 10s 开始，*FTP* 在第一个 *Remote Login* 应用结束之后 10s 开始，*Web* 在 *FTP* 应用结束之后 5s 开始，第二个 *Remote Login* 应用在 *Web* 应用终止之后 10s 开始。这个开始偏移时间代表开始一个应用可能需要的额外时间，但在许多情况中可能是不必要的。

概要的重复间隔时间被设置为 3600s 或 1h，代表 Alice 的 4h 班之间的休息时间。概

要被设置为在 1h 休息之后重复一次以上，这使在仿真过程中总共要执行两个概要会话。在概要开始时，所有应用都被配置为仅重复一次，确保每项应用在概要内不会重复。第一个 *Remote Login* 应用被设置为持续 5400s 或 1.5h，*FTP* 被设置为运行 600s 或 10min，*Web* 应用被配置为执行 1800s 或 30min，而第二个 *Remote Login* 应用被配置为运行 6600s 或 1h 50min。

最后，为在仿真时长期间确保两个概要执行所有它们配置好的应用，将仿真长度设置为 10h 35min（即 Jon 概要执行 10h 30min，为初始化和其他可能的延迟赋予额外的 5min）。图 7.10 给出 Jon 和 Alice 的概要配置一览，而图 7.11 给出仿真和这两个概要之间的关系。图 7.10 和图 7.11 没有按尺度画，仅说明应用、概要和仿真之间的关系。

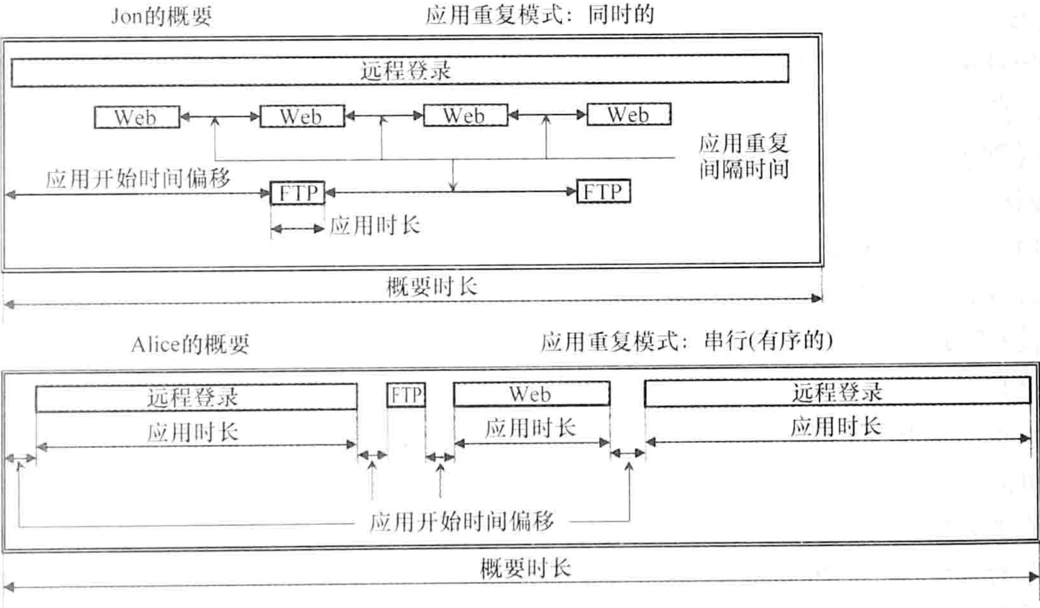


图 7.10　Jon 和 Alice 的概要

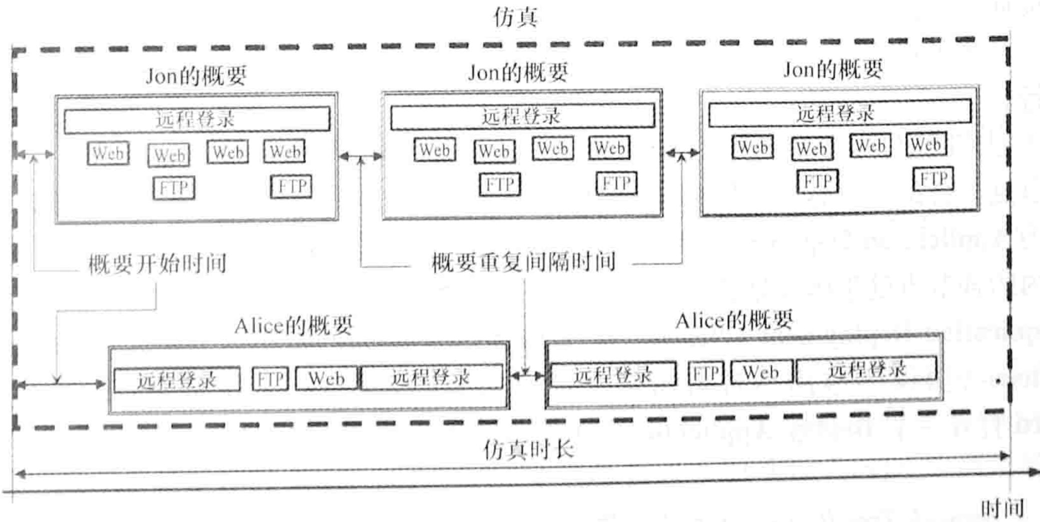


图 7.11　Jon 和 Alice 的概要与仿真之间的关系

注意多少有点复杂的应用使用行为可能需要通过多个用户概要来表示。例如，考虑这样一种情况，其中一名用户在一定时段同时运行几个应用，之后顺序地执行更多的应用。这样一个模式不能表示为单个用户概要，原因是在该概要内的各应用仅同时运行或串行运行，不允许更复杂的行为。为了配置这种类型的用户行为，可能希望创建两个独立的用户概要：第一个概要表示并行执行的应用，第二个概要表示顺序执行的应用。尽管如此，由 OPNET 提供的用户概要工具是非常灵活的，并可表示各种应用使用行为，这对多数仿真研究而言是足够用的。

7.4 使用 Application Deployment Wizard 部署用户概要

一旦定义了所有期望的应用和用户概要，则下一步是在被仿真系统内部署所定义的应用。典型情况下，在客户端—服务器范型中，应用部署包括指定将启动应用执行的各节点（即客户端）和将提供应用服务的各节点（即服务器）。使用 OPNET 术语，在被仿真网络系统中部署应用，要求如下步骤：

1）指定支持所定义应用概要的节点（即客户端或源）。

2）指定支持应用服务的节点（即服务器）。

在 OPNET 中，工作站、服务器、LAN 和负载均衡器对象都在对象内包含指定所支持概要和应用服务的配置属性。典型情况下，工作站、LAN 和负载均衡器对象作为支持所配置用户概要的客户端或源节点，虽然在一些情况中，在服务器节点部署用户概要也是可能的。类似地，服务器节点通常作为服务器，而在某些情形中（例如，当运行话音或视频应用时），将工作站对象配置为支持特定应用服务也是可能的。LAN 对象也可以作为服务器，原因是每个 LAN 对象假定由几个工作站和一个服务器节点组成。如果一个客户端节点支持某个概要，那么在那个概要内的每个应用都可在仿真过程中被执行。对服务器节点没有这样的约束。服务器不是直接绑定到一个特定概要的。相反，服务器负责支持特定应用。一般而言，人们期望的是，是在所部署概要中的每个应用至少支持一个服务器节点。如果没有支持一项特定应用服务的服务器，那么那个应用将从不会执行。

典型情况下，应用部署要求单独地配置各节点以支持期望的概要和应用。一次配置一个节点，会是一项容易出错和耗时的任务。这就是为什么 OPNET 产品的较新版本包含称为 **Application Deployment Wizard**（应用部署引导）的一个独立软件工具，支持在系统内的各节点处集体性地部署所指定的应用。为了从 **Project Editor** 的下拉菜单中打开 **Application Deployment Wizard**，选择 **Protocols→Applications→Deploy Defined Applications**（协议→应用→部署所定义的应用）。当启动时，**Application Deployment Wizard** 打开一个 **Deploy Applications** 窗口，该窗口由如图 7.12 所示的四个不同平板组成：

1）*Network Tree Browser*（网络树浏览器）平板控制在 **Deploy Applications** 窗口中来自被仿真系统的对象的显示。

2）*Application Deployment Operations*（应用部署操作）平板控制在概要内作为源或服务器的一个被选择节点的添加或去除，选择要被执行的应用部署操作的类型，并显示相应的部署视图。

3）*Applications Deployment Hints*（应用部署提示）平板，基于当前所选择的应用部署操作，就如何配置应用部署，提供有帮助的提示信息。

4）*Dialog Box Controls*（对话框控制）平板操作 **Deploy Applications** 窗口，并提供调试应用部署的功能特征。

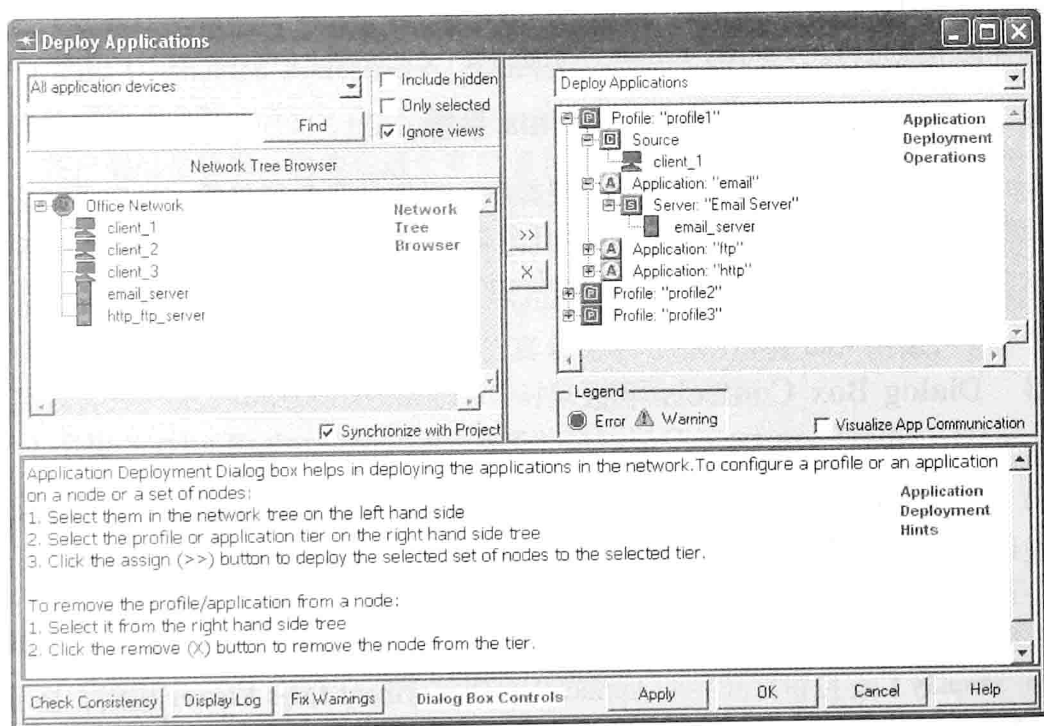


图 7.12　**Deploy Applications** 窗口

7.4.1　Network Tree Browser 平板

Network Tree Browser（网络树浏览器）平板在被仿真网络系统内显示可用的对象，同时在本节中的控制允许对要显示哪些对象进行定制。

1）在平板顶部的一个下拉菜单，指定要显示的节点类型。默认情况下，所有应用设备显示在 *Network Tree Browser* 中。但是，可选择限制仅显示 *workstation*、*servers*、*mainframes*（主机）、*LAN* 或 *load balancer*（负载均衡器）对象。

2）旁边有 **Find**（寻找）按钮的一个文本框，允许对在其名字中包含指定文本的节点进行搜索。在 **Project Editor** 中将选中识别出的节点，且如果选择框 *Synchronize with Project*（与项目同步）被选中，那么这些节点也将在 *Network Tree Browser* 中高亮显示。

3）另外，存在实现如下功能特征的四个检查框：

① *Include hidden*（包括隐藏的）。当选上时，*Network Tree Browser* 也将显示没有在

Project Editor 内给出的对象。

② *Only selected*（仅是选中的）。当选上时，*Network Tree Browser* 将仅显示在 **Project Editor** 内高亮显示的对象。仅当也选择 *Synchronize with Project* 检查框时，这个功能特征才起作用。

③ *Ignore views*（忽略视图）。当选上时，不管网络拓扑的配置视图为何，*Network Tree Browser* 将显示仿真模型中的所有对象。

④ *Synchronize with Project*（与项目同步）。当选上时，*Network Tree Browser* 和 **Project Editor** 被持续更新，从而它们的视图是一致的，即当在 *Network Tree Browser* 中选择节点时，同样的节点也将在 **Project Editor** 中被选中，反之亦然。

7.4.2 Application Deployment Hints 平板

Application Deployment Hints（应用部署提示）平板是非常简单的，且它不包含可操作的控制。这个平板为执行各种应用部署任务显示帮助信息。基于 **Deploy Applications** 窗口（见 7.4.4 节）中的当前选择，*Application Deployments Hints* 平板中的文本发生变化。

7.4.3 Dialog Box Controls 平板

Dialog Box Controls（对话框控制）平板由如下所述的 7 个按钮组成：

1）**Check Consistency**（检查一致性）按钮检验概要已经被正确地部署。如果一致性检查识别出应用部署中的任何错误，那么 *Application Deployment Hints* 平板将显示应用部署一致性检查失败的原因，它也将建议可能的解决方法。例如，一致性检查将识别出这样一种情况，其中没有服务器节点支持部署在客户端节点的概要中的应用。

2）**Display Log** 按钮打开一个 **Application Deployment Logs Viewer** 应用，显示一个错误/警告消息列表和解决当前应用部署配置问题的各项操作的描述。这个日志查看器应用包括如下功能特征，如将日志消息输入到一个电子表格、逗号分隔的文本文件、Web 报告或 XML 文件，以及管理日志消息将如何显示。

3）**Fix Warnings**（修正警告）按钮解决当前用户概要部署中的可能配置问题，并将执行的各项操作保存到一个日志文件。

4）**Apply** 按钮实施应用部署一致性检查。如果各概要被正确部署，则它在不关闭 **Deploy Applications** 窗口的条件下保存配置变化。

5）**OK** 按钮也实施应用部署一致性检查。如果各概要被正确部署，那么它保存配置变化，同时关闭 **Deploy Applications** 窗口。

6）**Cancel** 按钮在不保存配置变化的条件下，关闭 **Deploy Applications** 窗口。

7）**Help** 在网页浏览器中打开带有一个帮助页的窗口，它提供 **Deploy Applications** 窗口功能特征的一个简短描述。

7.4.4 Application Deployment Operations 平板

Application Deployment Operations（应用部署操作）平板的中间显示所配置概要的当

前客户端/服务器指派关系，将之称为 *Profile Definition Tree*（概要定义树）。但是，*Profile Definition Tree* 的实际外观取决于 *Application Deployment Operations* 平板顶部下拉菜单中的值。这个下拉菜单控制要实施的部署操作的类型，有 4 个可能选项：

1）**Deploy Applications**（部署应用）通过为每个概要指定客户端和服务器节点，在被仿真网络中部署应用。当选中这个选项时，*Profile Definition Tree* 将在依据用户概要而组织的网络中显示当前应用部署。概要层是顶层，且它包含单个概要的应用部署。每个概要层包含单个源层和一个或多个应用层，一层对应于概要内定义的每个应用。最后，每个应用层也包括一个服务器层。

2）**Edit Destination Preferences**（编辑目的地首选项）支持配置服务器选择权重。当这个选项被选中时，*Profile Definition Tree* 将依据客户端节点显示应用部署。在这个视图中，客户端层是顶层。它为在这个客户端节点处部署的概要中定义的每项应用，包含一个独立层。最后，每个应用层包含支持这项应用的服务器列表。

3）**Edit LAN Configuration**（编辑 LAN 配置）支持在 LAN 对象内指定部署的概要客户端的数量。当这个选项被选中时，*Profile Definition Tree* 将依据 LAN 对象的排列而显示应用部署。该树的顶层将列出在被仿真系统中可用的所有 LAN 对象，且每个 LAN 对象层包含另一层，其中列出在相应 LAN 对象处部署的概要。

4）**Edit Service Registration**（编辑服务注册）和 **Convert Ace Traffic**（**Flows < - > Discrete**）[转换 Ace 流量（流 < - >离散的）] 选项处理 OPNET 的 ACE 软件，这里不做讨论。

图 7.13 给出下拉菜单不同值时 *Profile Definition Tree* 的外观。另外，*Application Deployment Operations* 平板包含一个 *Visualize App Communication*（虚拟化应用通信）检查框，当点选时打开一个 **Application Communication Visualization**（应用通信虚拟化）窗口。在 *Network Tree Browser* 和 *Application Deployment Operations* 平板之间也有两个按钮："> >" [称作 **Deploy Node to a Tier**（将节点部署到一层）] 和 "**X**" [称作 **Remove Node from a Tier**（从一层中去除节点）]。

在 7.4.5 ~ 7.4.7 节比较详细地描述 **Deploy Applications**（部署应用）、**Edit Destination Preferences**（编辑目的地首选项）和 **Edit LAN Configuration**（编辑 LAN 配置）选项。

7.4.5　Deploy Applications 选项

当选中 **Deploy Applications**（部署应用）选项时，通过实施如下动作，可在所定义的用户概要中指派节点作为客户端和服务器（即部署应用）：

1）从 *Application Deployment Operations* 平板的下拉菜单中，选择 *Deploy Application* 选项（当 **Deploy Applications** 窗口被打开时，这是默认选项）。

2）展开 *Profile Definition Tree*，从而使源、应用和服务器层是可见的。

3）指派一个客户端或源节点：

① 在 *Network Tree Browser* 中选择一个节点。

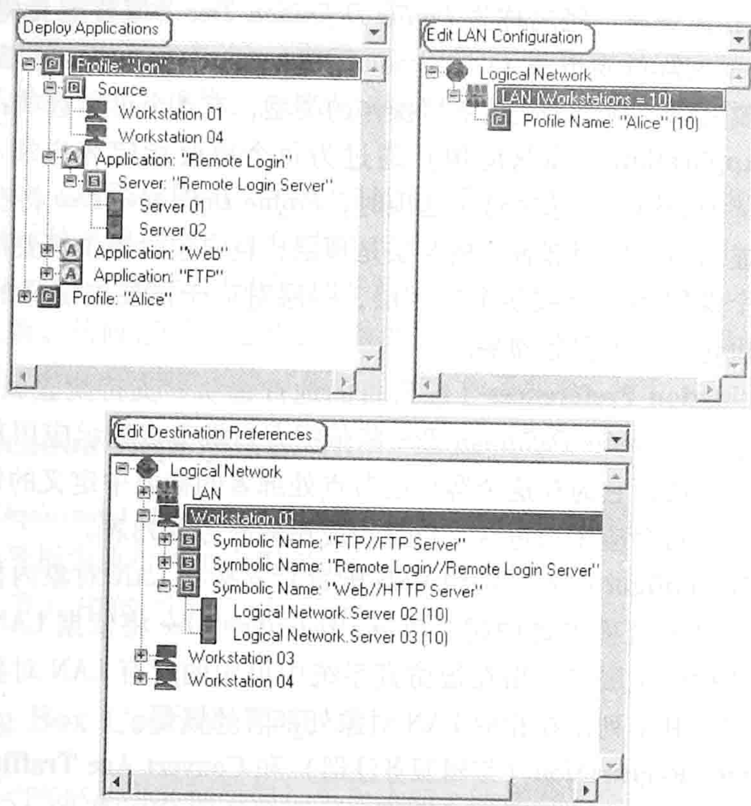


图7.13 *Profile Definition Tree*（概要定义树）

② 在*Profile Definition Tree*中选择期望概要的源层。

③ 单击“>>”按钮，将节点指派到所选择的源层。这将使*Network Tree Browser*中的节点图标出现在源层。注意，所指派的节点将作为当前概要内所有应用的一个源或发起点（initiator）。

④ 如有必要，重复这个过程。单个概要可有多个客户端节点，这意味着在仿真过程中每个被指派的节点将运行相应的概要。

4）指派一个服务器节点：

① 在*Network Tree Browser*中选择一个节点。

② 在*Profile Definition Tree*中选择期望概要的服务器层。注意每个服务器层与概要内一个特定应用相关联。

③ 单击“>>”按钮，将节点指派到所选择的服务器层。这将使所选择的节点的图标出现在服务器层图标的下面。所添加的节点将作为包含所选择应用的所有概要中该应用的一个服务器。注意一个给定节点可作为多个应用的服务器。

④ 如有必要，重复这个过程。单个应用可得到多个服务器节点的支持，即运行一个应用的客户端节点，可连接到支持那个应用的服务中的任何一个服务器。

为了从*Profile Definition Tree*中作为一个客户端或服务器的当前指派中去除一个节点，实施如下动作：

1）在*Profile Definition Tree*中期望概要的一个源或服务器层下，选择一个节点。

2）单击“X”按钮，从所选择源或服务器层去除该节点。

一旦指派了所有的客户端和服务器节点，应用部署就是完全的，且就可单击 **Apply** 或 **OK** 按钮，保存应用部署配置。

一个客户端节点可连接到任何一台服务器，该服务器在客户端节点上支持一项应用（即客户端节点支持的概要的组成部分）。通过一个服务器选择权重属性，OPNET 控制一个特定服务器选择的概率。服务器选择权重是客户端节点的属性，即它们定义在应用执行期间客户端节点连接到一个特定服务器的概率。那就是当选择 **Edit Destination Preferences** 选项时 *Profile Definition Tree* 依据客户端节点而组织排列的原因。因此，不同的客户端节点可针对同一服务器节点设置不同的服务器选择权重值。默认情况下，所有服务器都被指派为 10 个单位的相同权重，结果是，选择一个特定服务器的概率是相同的。服务器选择权重的较大值，增加了该服务器将被客户端节点选择的概率[⊖]。

考虑如下例子。假定有三个服务器节点，称为 *S1*、*S2* 和 *S3*，支持 FTP 应用。同样，假定客户端节点 *C1* 和 *C2*，支持包括 FTP 应用的一个用户概要。假定节点 *C1* 针对服务器 *S1*、*S2* 和 *S3* 分别将服务器选择权重设置为 10、15 和 25，而节点 *C2* 配置这些选择权重为 25、40 和 35。因此，在仿真过程中，节点 *C1* 将以概率 10/(10+15+25)=0.20 连接到服务器 *S1*，以概率 0.30 连接到服务器 *S2*，以概率 0.50 连接到 *S3*。类似地，节点 *C2* 将分别以概率 0.25、0.40 和 0.35 选择服务器 *S1*、*S2* 和 *S3*。

最后，仅当在 **Application Deployment Operations** 平板的下拉菜单中选择 **Deploy Applications** 选项时，*Visualize App Communications* 检查框才是可见的。当选上时，打开 **Application Communication Visualization** 窗口，这将提供如图 7.14 所示的应用配置的一个视觉地图（visual map）。这个窗口包含三个下拉菜单：

1）第一个菜单包含在相应用户概要内定义的应用列表。

2）第二个菜单指定在被仿真系统中定义的概要列表。

3）第三个菜单提供支持当前所选择的用户概要的客户端节点列表和仅显示逻辑层的一个选项。

改变下拉菜单中的选择，就改变在 **Application Communication Visualization** 窗口中显示的通信图片。图 7.14 显示，*FTP* 应用是 Jon 的概要的组成部分（该概要部署在一个名为 *LAN* 的对象处，具有一对一的客户端—服务器逻辑通信），且在仿真过程中，在 *LAN* 节点上执行的 FTP 应用，连接到称为 *Server 04* 和 *Server 03* 的两个服务器节点。

7.4.6　Edit Destination Preferences 选项

从 *Application Deployment Operations* 平板的下拉菜单中选择 **Edit Destination Preferences**（编辑目的首选项）选项，允许改变每个客户端节点的服务器权重。当选择这个选项时，只能在那些节点上改变服务器（被配置为支持概要的应用）权重，不允许添加新的服务器节点。改变服务器选择权重的步骤如下：

⊖ 外部 C 代码文件 **tpal_ app_ support. ext** 包含称为 ***app_ server_ name_ select*** **( )** 的一个函数，该函数基于一个服务器的权重随机地选择服务器。

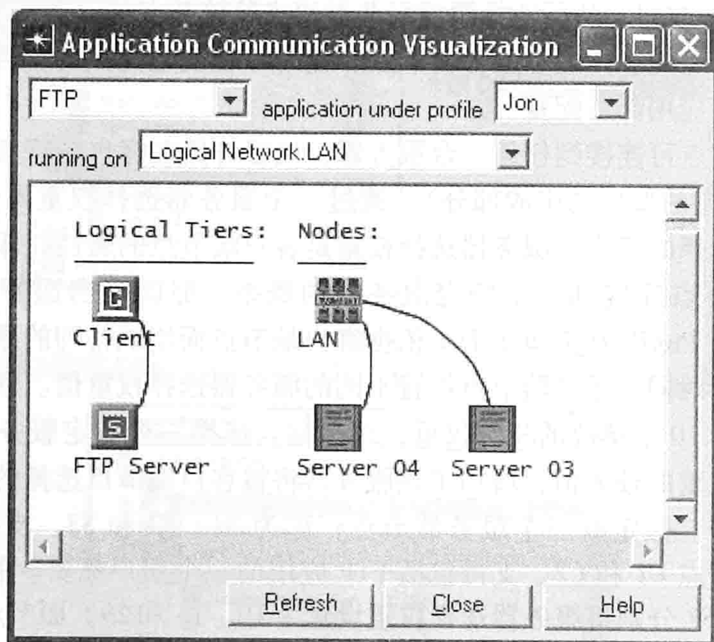


图 7.14 **Application Communication Visualization**（应用通信可视化）窗口

1）从 *Application Deployment Operations* 平板的下拉菜单中，选择 **Edit Destination Preferences** 选项。

2）展开 *Profile Definition Tree*，从而客户端及其相应的服务器层是可见的。

3）选择一个期望的服务器层或在那个层中的一个实际服务器节点。

4）单击 **Edit** 按钮，或双击服务器或服务器节点层，打开 **Edit Destination Preferences** 窗口，该窗口包含支持当前应用的所有服务器的一个表。

5）在 **Edit Destination Preferences** 窗口中，单击 **Weight** 列，改变服务器选择权重值。如果针对一个特定应用的所有服务器权重都被设置为 0，那么将以相同的概率选择所有服务器。但是，如果至少有一个服务器有一个非零的权重，那么权重为 0 的所有服务器都将不被选择。如图 7.13 所示，当选择 **Edit Destination Preferences** 选项时，服务器权重出现在每个服务器层旁边的括号中。

6）单击 **OK** 按钮关闭 **Edit Destination Preferences** 窗口并保存变化，或单击 **Cancel** 按钮，在不保存变化的情况下关闭窗口。

7）如果期望的话，针对其他的客户端节点，重复上述过程。

7.4.7 Edit LAN Configuration 选项

Edit LAN Configuration（编辑 LAN 配置）选项支持在一个 LAN 对象内指定部署的概要客户端数量。当选择这个选项时，只有支持用户概要的那些 LAN 对象才在 *Profile Definition Tree* 中是可见的。所有其他类型的对象和不支持任何用户概要的 LAN 对象都不会是可见的。改变一个 LAN 对象内所部署的客户端数量的部署如下：

1）从 *Application Deployment Operations* 平板的下拉菜单中选择 **Edit LAN Configuration**

选项。

2）展开 *Profile Definition Tree*，使包含 LAN 对象的层及其概要是可见的。

3）选择期望的 LAN 对象或其概要层。

4）单击 **Edit** 按钮，或在 LAN 对象或其概要层上双击，打开 **Edit LAN Profile Configuration** 窗口，其中包含在当前 LAN 对象内部署的所有概要的一个表。

5）在 **Edit LAN Profile Configuration** 窗口中，单击 *Number of Clients* 列，改变 LAN 中概要客户端数量。OPNET 建议，不要将客户端数量配置为大于 LAN 中工作站的数量，虽然 OPNET 不会防止这样一种配置。如图 7.13 所示，如果选择 **Edit LAN Profile Configuration** 选项，则客户端数量和工作站数量将分别出现在概要和 LAN 对象层旁边的括号中。

6）单击 **OK** 按钮关闭 **Edit LAN Profile Configuration** 窗口并保存变化，或单击 **Cancel** 按钮，在不保存变化的情况下关闭窗口。

7）如果期望的话，针对其他的 LAN 对象，重复上述过程。

7.4.8　Clear Application Deployment 选项



最后，在一些情况下，与尝试识别配置问题的做法相比，完全清除当前应用部署是比较容易的。OPNET 提供一个工具，可从当前仿真场景中清除所有的应用部署。为了清除应用部署，从 Project Editor 的下拉菜单中选择 **Protocols→Applications→Clear Application Deployment**（协议→应用→清除应用部署）选项。结果是，**Clear Application Deployments** 窗口将出现。这个窗口由三个检查框和标准控制按钮（**OK**、**Cancel** 和 **Help**）组成。检查框允许配置要清除的应用部署类型：

1）*ACE traffic flow deployment*（ACE 流量断续流部署）——清除通过 ACE OPNET 软件产品（这里不做讨论）所定义的所有流量部署。

2）*DES application demand deployment*（DES 应用要求部署）——清除定义为应用要求（见 6.6.1 节）的所有流量部署。

3）*DES application deployment*（DES 应用部署）——通过将相应对象的属性重置为其默认值，清除在客户端和服务器节点（即工作站、服务器、LAN、负载均衡器和主机对象）上的所有概要和应用部署。不管应用部署是否采用 **Application Deployment Wizard**，应用部署都将被清除。这个动作将不修改或去除概要、应用和应用任务定义。

控制按钮具有同样的标准含义：

1）**OK**——基于检查框选择，清除应用部署，并关闭当前窗口。

2）**Cancel**——在不实施任何动作的条件下，关闭当前窗口。

3）**Help**——打开一个帮助页面，带有在这个窗口中存在功能特征的简短描述。

总之，实施如下动作，清除在当前场景中的应用部署：

1）从 **Project Editor** 的下拉菜单中选择 **Protocols→Applications→Clear Application Deployments** 选项。

2）在出现的 **Clear Application Deployments** 窗口中，选中对应于要被清除的应用

部署类型的检查框。默认情况下，所有检查框都是被选中的，这是足够用的且可不做改变，原因是OPNET仅清除被配置的那些应用部署类型。

3）单击 **OK** 按钮清除选中的应用部署或单击 **Cancel** 按钮撤销这个动作并关闭相应的窗口。

7.5 在不使用 Application Deployment Wizard 条件下部署用户概要

在不使用 **Application Deployment Wizard** 的条件下，也可在被仿真的网络系统内部署应用。典型情况下，这个过程涉及指定被仿真网络中的哪些端节点作为客户端（通过支持所定义的用户概要）和哪些端节点作为服务器（通过为所定义的应用提供服务）。本节讨论在不用 **Application Deployment Wizard** 的条件下，与部署所定义应用有关的这些问题和其他问题。

7.5.1 配置客户端节点

典型情况下，通过实施如下步骤，将客户端节点配置为支持一个定义好的用户概要：

1）右击关注的客户端节点，并选择 **Edit Attributes** 选项。

2）展开属性 **Applications**。

3）展开属性 **Applications：Supported Profiles**。

4）将属性 **Number of Rows** 属性的值设置为希望在当前客户端节点处部署的概要数量。

5）对于创建的每一行：

① 展开行。

② 设置 **Profile Name** 属性的值。这个属性使您可选择已经在仿真中配置的概要之一。不允许其他值。如果当前仿真场景没有定义好的概要，那么 **Profile Name** 属性的值将为 None（无），且不能部署概要。一旦设置 **Profile Name** 的值，则相应行的标题也改变为指定的值。

③ 可以不改变其他属性。注意，通过 **Edit Attributes** 菜单，属性 **Traffic Type** 是不可配置的。通过 **Application Deployment Wizard**，可改变这个属性的值。这个属性处理OPNET的ACE软件，在本书中不做讨论。属性 **Application Delay Tracking** 配置这样一种机制，即识别离散事件仿真中的应用延迟的源，本书中也不做讨论。

④ 单击 **OK** 按钮，保存所做的配置改变。

对于支持所定义的用户概要的所有客户端节点对象，应该重复这个过程。在由许多客户端节点组成的一个仿真中，一次配置一个节点的做法，是烦琐的和容易出错的任务。为了加快应用部署，可使用 **Application Deployment Wizard**，或如果多个客户端运行相同的概要集合，那么它们可作为一个进行配置（即选择期望的对象，在所选中对象之一中编辑属性，并在单击 **OK** 按钮之前，单击 **Edit Attributes** 窗口底部处的 **Apply**

to Selected Objects 检查框)。图 7.15 给出在一个客户端处概要部署的一个例子。

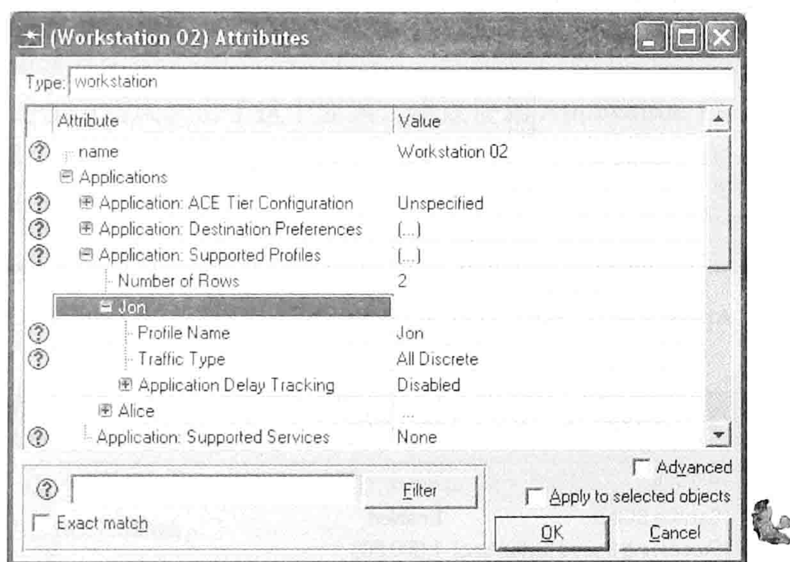


图 7.15　在一个客户端处的概要部署

7.5.2　配置服务器节点

在服务器节点部署应用服务也是非常简单的，可通过执行如下步骤做到这一点：

1）右击关注的服务器节点，并选择 **Edit Attributes** 选项。

2）展开属性 **Applications**。

3）单击属性 **Applications：Supported Services**，这会打开如图 7.16 所示的 **Application：Supported Services Table** 窗口。

① 通过设置在表窗口底部处 **Rows** 的值，指定在这个节点处所支持的应用数量。一旦设置该值，则相应的行数将出现在表中。注意将行数设置为大于这个场景中 *Application Config* 节点中定义的总应用数，将导致一些行不被设置，原因是该表不能包含具有相同名字的两行。

② 通过从下拉菜单中选择一个应用名，指定在这个服务器处支持的应用。一旦已经选择一个特定应用，则其名字将不会再次出现在那个表后续行的选择中。

③ 在描述列中，选择 Supported、Not Supported 或 Edit...。当值 Supported 被选择时，那么应用将使用默认设置由当前服务器支持。如果值 Not Supported 被选中，那么应用将得不到这个节点的支持，且运行这个应用的客户端节点将不能连接到那个应用各项服务的当前服务器。如果值 Edit... 被选中，那么打开一个表，允许配置服务器的应用支持属性（即处理速度、每条请求的额外负荷延迟、ToS 值等）。图 7.17 给出这样一个表的默认设置。

④ 单击 **OK** 按钮保存所做的改变。

4）另外，**Application：Supported Services** 有两个预设值：None（无）（意味着当

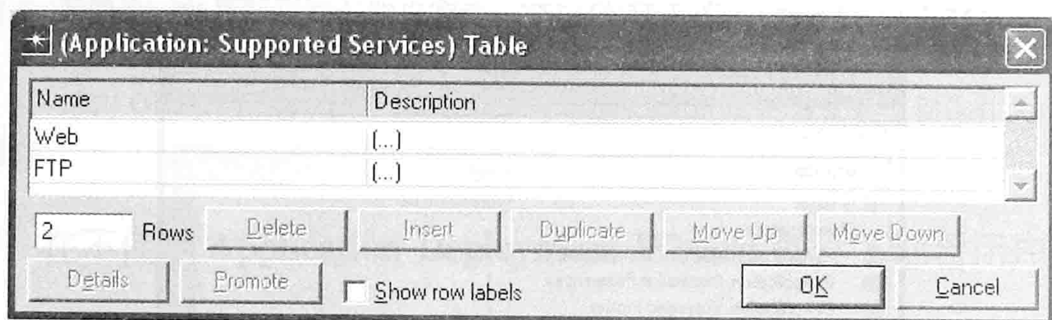


图 7.16 **Application: Supported Services Table**（应用：支持的服务表）窗口

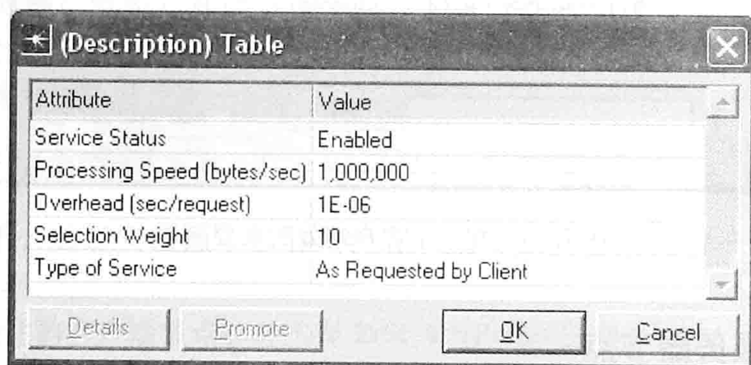


图 7.17 所支持应用服务的默认配置

前服务器不支持任何应用服务）和 All（所有的）（意味着这个服务器支持所有应用服务）。OPNET 建议，不要将服务器节点设置为支持所有应用服务，因为如果在后来配置发生改变且定义新的应用，那么这个节点会支持仿真并不打算支持的应用。事实上，**Application Deployment Wizard** 将这样一种配置处理为一个警告，以一个告警符号标记相应的节点。

7.5.3 指定目的地首选项

回顾一下，每个应用定义（即在 *Application Config* 节点中）包含指定符号服务器名的一个属性。存在于所有客户端节点中的属性 **Application: Destination Preferences**，配置服务器选择权重，并提供仿真中符号服务器名和节点对象的实际名之间的一个映射。在这个属性的帮助下，每个个体客户端节点可指定可能服务器目的地的一个不同集合且为每项应用指定服务器选择权重。如果 **Application: Destination Preferences** 保持不变，那么默认情况下，OPNET 将支持应用服务的每个服务器节点配置为可能的目的地之一，服务器选择权重为 10（即相同的选择概率）。参见 5.5.2 节，了解指定目的地首选项的详细指令。

如您所想到的，这个过程是容易出错的，因为容易设置不正确的符号服务器名或将不支持当前应用的一个服务器节点配置为应用目的地之一（即下拉菜单列出系统中的所有节点，且不提供有关哪些节点支持当前应用哪些节点不支持当前应用的任何提

示）。如果目的地首选项进行了不正确的配置，那么 OPNET 仿真退回到默认设置，其中支持当前应用的每个服务器节点以相同的概率被选作一个目的地（即默认情况下，服务器选择权重被设置为 10）。这也许有点令人迷惑，原因是 OPNET 使用默认设置，但却不改变配置中的任何错误。出于这个原因，建议使用 **Application Deployment Wizard** 完成所有的应用部署，并仅当应用目的地首选项属性在 **Application Deployment Wizard** 中不存在时，才对其进行配置。

7.5.4 在一个 LAN 对象中指定客户端数量

Application Deployment Wizard 包含这样一项功能，它允许指定 LAN 对象中的客户端数量。在不使用 **Application Deployment Wizard** 的条件下，通过简单地执行如下步骤，可设置客户端数量：

1）右击关注的 LAN 对象，并选择 **Edit Attributes** 选项。

2）展开属性 **Application**。

3）展开属性 **Application：Supported Profiles**，并配置在这个 LAN 对象处支持的用户概要。

4）展开描述所关注概要的属性。

5）设置属性 **Number of Clients** 的值，它指定在 LAN 中运行当前概要的用户数。默认情况下，这个值被设置为 Entire LAN（整个 LAN），这意味着 LAN 中的每个节点都将运行这个概要的一个实例。OPNET 建议，不要将客户端数量设置为大于 LAN 中的工作站数量。

6）单击 **OK** 按钮关闭窗口并保持所做的改变。

也可如下指定一个 LAN 对象中的工作站数量：

1）右击所关注的一个 LAN 对象，并选择 **Edit Attributes** 选项。

2）展开属性 **LAN**。

3）设置属性 **Number of Workstations** 的值，指定在这个 LAN 中的工作站总数。假定每个 LAN 模型由许多工作站和单个服务器组成。

4）单击 **OK** 按钮关闭窗口并保存所做的改变。

图 7.18 给出在一个 LAN 对象中配置工作站数量和客户端数量的属性。

7.5.5 指定一个应用使用的传输协议

最后，在某些情况下，改变由一个应用使用的基础传输协议，可能是必要的。默认情况下，标准应用如数据库、电子邮件、FTP、HTTP、远程登录和打印都使用 TCP，而话音和视频会议则依赖于 UDP。仅在 LAN、负载均衡、主机和服务器及工作站对象的高级节点模型（即节点模型在其名字中有扩展名_ *adv*）中才可改变由应用利用的传输协议。为完成这项任务，实施如下步骤：

1）右击关注的一个节点模型，并选择 **Edit Attributes** 选项。

2）展开属性 **Application**。

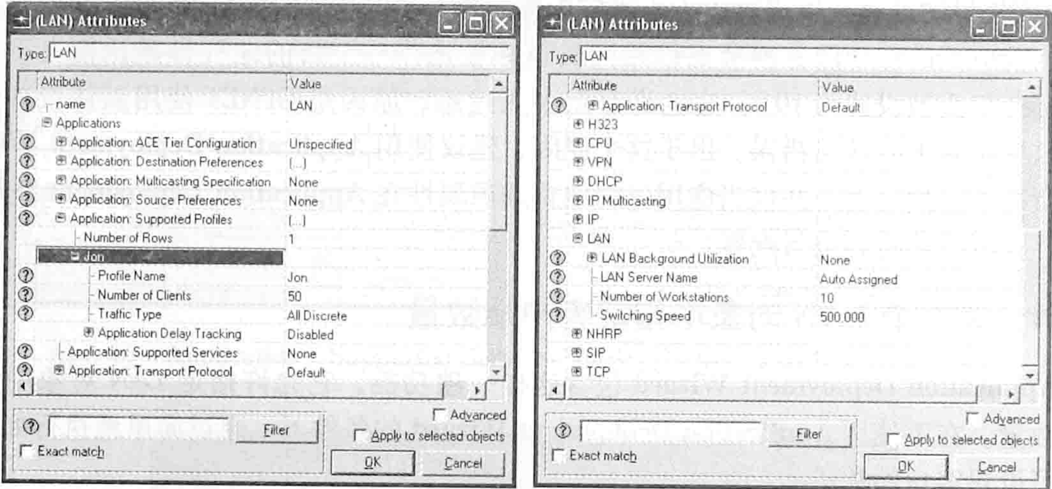


图 7.18　在一个 LAN 对象中指定客户端数量和工作站数量的配置属性

3）展开属性 **Application：Transport Protocol Specification**（应用：传输协议规范）。

4）为期望的应用改变传输协议。OPNET 提供如下传输协议选项：TCP、UDP、AAL5、X25、FR 和 NCP，虽然可用的协议随应用类型不同而变化。

为了使仿真正确地执行（即没有错误），必须在客户端和服务器节点上改变应用的传输协议。

7.6　在概要配置和应用部署中的常见错误

当配置用户概要时，必须了解应用、概要和仿真之间的相互作用，以便避免可能导致执行仿真时的不可预测结果或失败的各种错误。首先，详细研究与概要和仿真定时相关联的配置错误。

1）最常见的错误之一是将概要的时长设置为大于仿真的长度。在这种情形中，应用将在仿真相同的时间终止。例如，假定仿真的长度被设置为 300s，而概要被配置为运行 500s。如果概要被配置运行多个应用，那么就有可能在概要内的一些应用将不会启动。结果，相比预期的情况，将有较少的流量通过网络。

2）类似地，将概要重复间隔时间设置为大于仿真结束前剩余的时间，这将使概要不能重复，且同样会给出不可预料的结果。例如，假定仿真、概要和概要重复间隔时间的长度分别设置为 500s、300s 和 200s。如果概要被配置为在时刻 100s 处启动，那么概要将不会重复，原因是概要的第一个实例将在时刻 400s 处结束执行，而概要的第二个实例将在时刻 600s 处被调度执行，这已经处于仿真终止之后了。图 7.19 给出这种场景。

3）必须确保概要启动时间小于仿真的时长。例如，如果概要被配置为在时刻 400s 处启动，而仿真时长仅有 300s，那么概要将从不会执行。

4）将概要时长设置为 End of Simulation（仿真结束），将导致概要从不会重复执行。

第二种类型的概要配置错误是与概要内应用的不正确配置关联的，描述如下：

1）将应用启动偏移时间设置为大于概要的时长，将导致那个应用从不会启动。同样，如果概要被配置为以串行顺序执行应用，那么应用启动偏移时间加上前面各应用的启动偏移时间和时长，不应超过概要的长度，否则这个应用将从不会启动。图 7.20 给出这样一种情况。

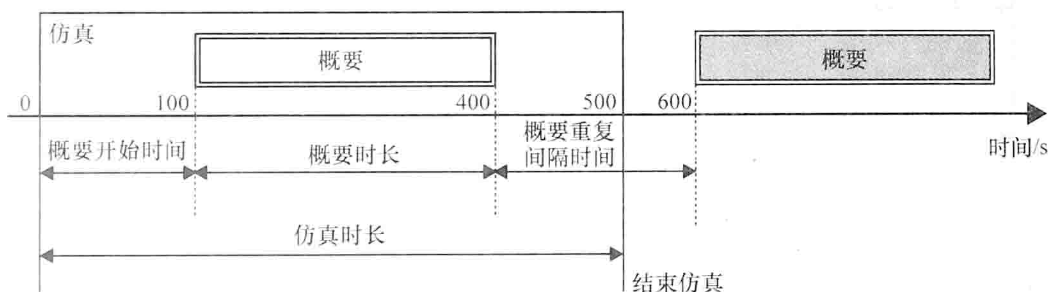


图 7.19 正确配置的 **Profile Inter – repetition Time** 例子

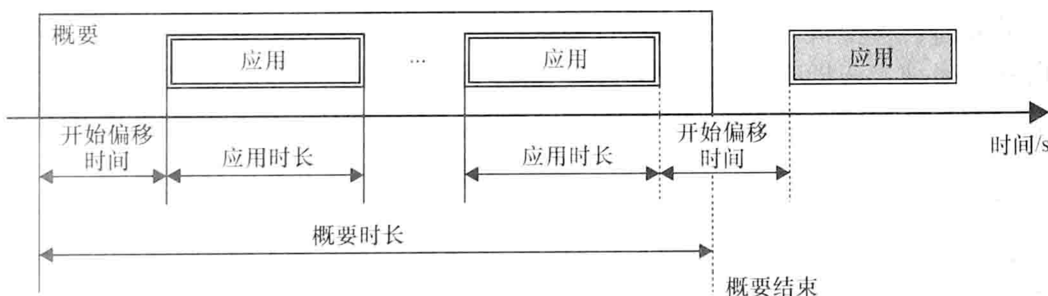


图 7.20 不良配置的 **Application Start Offset Time** 例子

2）类似地，如果概要启动时间和应用偏移时间的总和大于仿真时长，那么应用将从不会启动。OPNET 文档建议，这个问题常发生于应用偏移时间或概要启动时间的值使用一个指数分布配置时。

3）将应用时长设置为大于概要长度，将导致在网络中产生的流量低于预期的总量，并将使任何后续的应用不能启动（如果概要配置为以串行顺序执行应用的话）。

4）类似地，如果概要配置为以串行顺序运行应用，那么将一个应用的时长设置为 End of Simulation，将使所有后续应用不能启动。

5）如果概要配置为以串行顺序执行应用，将一个应用的可重复性设置为无限的，将使任何后续配置的应用不能启动。

一般情况下，当配置一个应用的重复间隔时间、启动偏移时间和时长时，人们应该小心从事。如果在概要内的各应用配置为以串行顺序执行，那么重要的是确保在应用完成其执行（包括所有的应用重复次数）之后，为所有后续定义的应用执行留下足够的时间。图 7.21 给出这样一种情形。另外，如果在概要内的各应用配置为同时执行，那

么对任意配置的应用，应用时长及其重复间隔时间之和不应该超过概要的时长，原因是如果超过的话，将使一个应用不能重复执行。这种情况类似于图 7.19 所示的情况。

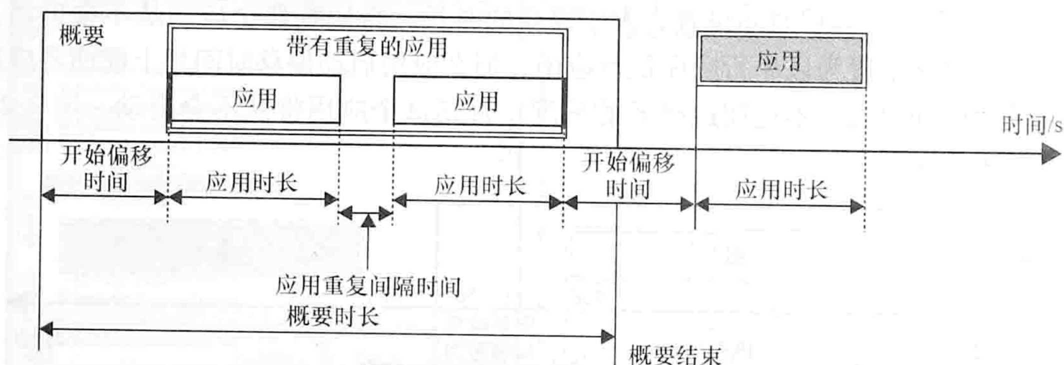


图 7.21　配置为执行得太长从而使后续定义的应用不能启动的应用例子

其他配置问题包括在一个客户端而不是在一个服务器上部署一个概要（相反情况亦然），或更一般的情况是，应用或概要配置的一些属性被忘记定义了。这种错误可能使仿真不能执行。最后，将各概要配置为仿真执行开始稍后启动，是一种好的想法。这个额外的延迟，为网络中其他协议进行初始化提供了一次机会。例如，将一个概要配置为在时刻 0 启动，并将应用偏移时间设置为 0，可能导致没有流量通过网络，原因是路由协议没有足够的时间更新和设置它们的路由表。在这种情形中，当应用报文到达一个节点的 IP 层时，没有路由到达目的地，这导致这些报文被丢弃。典型情况下，将概要启动时间设置为 100s，为所有协议进行初始化并收敛到其稳定状态提供了足够的时间。

一般情况下，当调试应用和概要配置或被仿真系统的任何其他部分时，可能希望利用如下的一条或多条建议：

1）使用 DES 日志，这提供了带时间戳的警告和错误消息的一个列表，其中有问题可能原因的一个解释。从 **Project Editor** 的下拉菜单中通过选择 **DES→Open DES Log**（DES→打开 DES 日志），可访问 DES 日志。欲了解更多细节，见 4.8 节。

2）当指定各种属性值时，通过使用常数分布函数，降低仿真的随机性。

3）考虑使用 *All Values*（所有值）模式收集统计量（如发送的流量和接收的流量），得到有关应用性能的更准确反馈。

4）通过降低被仿真网络系统的尺寸和简化整体配置，尝试隔离问题。

5）可通过 **Scenarios** 下拉菜单，考虑使用 OPNET 产生的报告［如 **User - Defined Reports**（用户定义的报告）和 **Object/Attribute Difference Report**（对象/属性差异报告）］。

第 8 章　传输层——TCP 和 UDP

8.1　引言

互联网传输层（层 4）包含两个广泛使用的协议：**Transmission Control Protocol**（**TCP**，传输控制协议）和 **User Datagram Protocol**（**UDP**，用户数据报协议）。TCP 是一个面向连接的可靠协议，在端节点系统上的进程之间提供有序的数据交付。在 TCP 中，一个 segment（分段）是带有一个 TCP 首部的数据单元，该首部是在两个 TCP 端口实体之间传递。TCP 跟踪已经发送但还没有成功确认的分段，并重传丢失的任何分段。典型情况下，TCP 由诸如电子邮件、远程登录和 FTP 等应用使用，都要求可靠的数据交付。另外，UDP 是一个无连接的和不可靠的协议，就成功的数据交付方面不做任何保障。UDP 简单地将数据交给低层网络层，希望数据将被交付到目的地进程。UDP 不跟踪所传递的数据包，就数据丢失的情况不做任何事情。UDP 主要由如下情况下的应用所用，其中快速数据交付（即低延迟）优先于偶然的报文丢失和缺乏有保障的交付情况。IP 上的话音（VoIP）和视频会议是运行在 UDP 上的典型应用范例。

OPNET 支持 TCP 和 UDP。但是，UDP 是一个非常简单的协议，且没有任何配置属性。另外，TCP 是十分复杂的，并在端节点模型上有大量可配置的属性，如服务器、工作站和 LAN 对象。典型情况下，端节点模型在其名字中包含 wkstn、server 或 LAN。核心节点模型（如集线器、交换机和路由器）不包含 TCP 配置属性，原因是这些节点模型不支持层 4 功能。即使 UDP 不包含任何配置属性，也可指定一个应用是运行在 TCP 之上还是 UDP 之上。默认情况下，OPNET 在 TCP 之上运行所有标准应用。仅有的例外是视频会议和话音，是配置为在 UDP 之上运行的。可改变默认的传输协议设置，方法是修改节点属性 **Application... Application：Transport Protocol Specification**（应用...应用：传输协议规范）的值，如图 8.1 所示。这个属性仅存在于高级节点模型中（即模型名包含扩展名_ *adv*）。也可参见 7.5.5 节，了解一个应用如何改变传输协议。

本章后面部分专门讨论 TCP 所支持的功能特征和可用的配置参数。具体而言，在 8.2 节，讨论 OPNET 支持 TCP 功能和风格（flavors），在 8.3 节中后跟常用 TCP 配置参数的一个概述。以 8.4 节收尾，该节描述监测 TCP 和 UDP 性能的可用统计量。

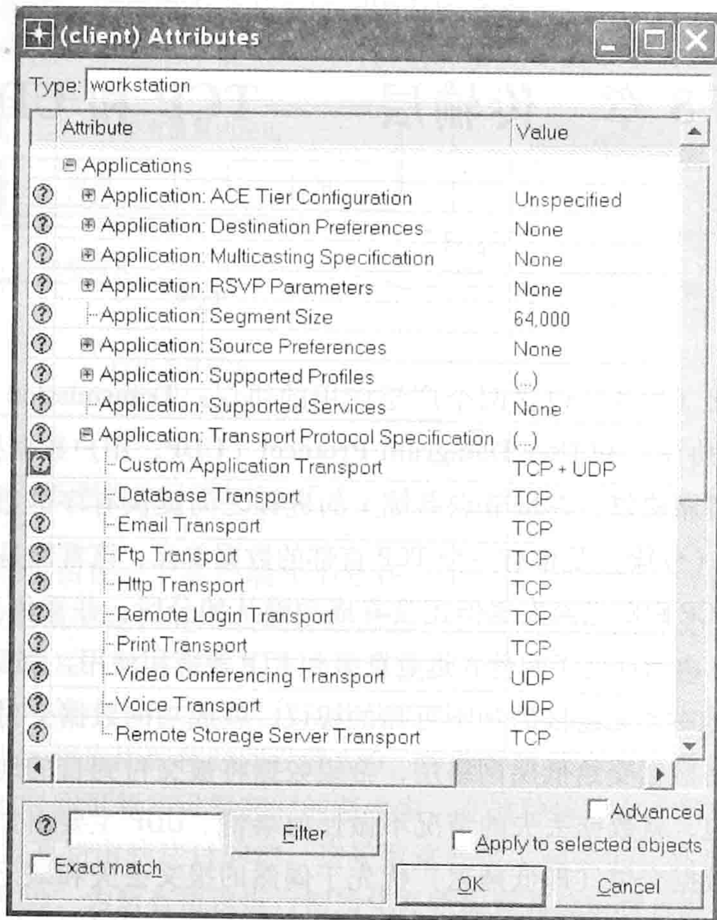


图 8.1 属性 **Application：Transport Protocol Specification**

8.2 支持的 TCP 功能

OPNET TCP 模型实现了所有的基本 TCP 功能，包括连接建立的三次握手算法、成功接收分段的确认、被认为丢失的分段的重传、动态窗口控制、有序数据交付（即在分段被发送到上层之前，乱序到达的各分段被重新排序）、各种拥塞管理机制、四次连接关闭过程以及其他。OPNET 也支持 TCP 的慢启动和拥塞避免机制、Nagle 的算法、Karn 的算法、快速重传和快速恢复、窗口缩放、选择性确认（SACK）、显式拥塞通知（ECN）、持久超时等。通过 **TCP... TCP Parameters**（TCP... TCP 参数）属性可配置这些功能特征，如图 8.2 所示。**TCP Parameters** 是一个复合属性，包括所有 TCP 相关的属性。为了在一个节点上指定 TCP 配置，可配置个体 TCP 配置属性，或为 **TCP Parameters** 属性选择一个预设值。每个预设值代表对应于特定 TCP 风格的 TCP 配置或架构特定的 TCP 设置。图 8.2 列出 **TCP Parameters** 属性的所有预设值。

每种 TCP 风格（如 Tahoe、Reno、New Reno、SACK 等）定义了 TCP 拥塞控制机制的一个特定配置。代表一个架构特定设置的一个预设值，定义了对应于一个特定平台上

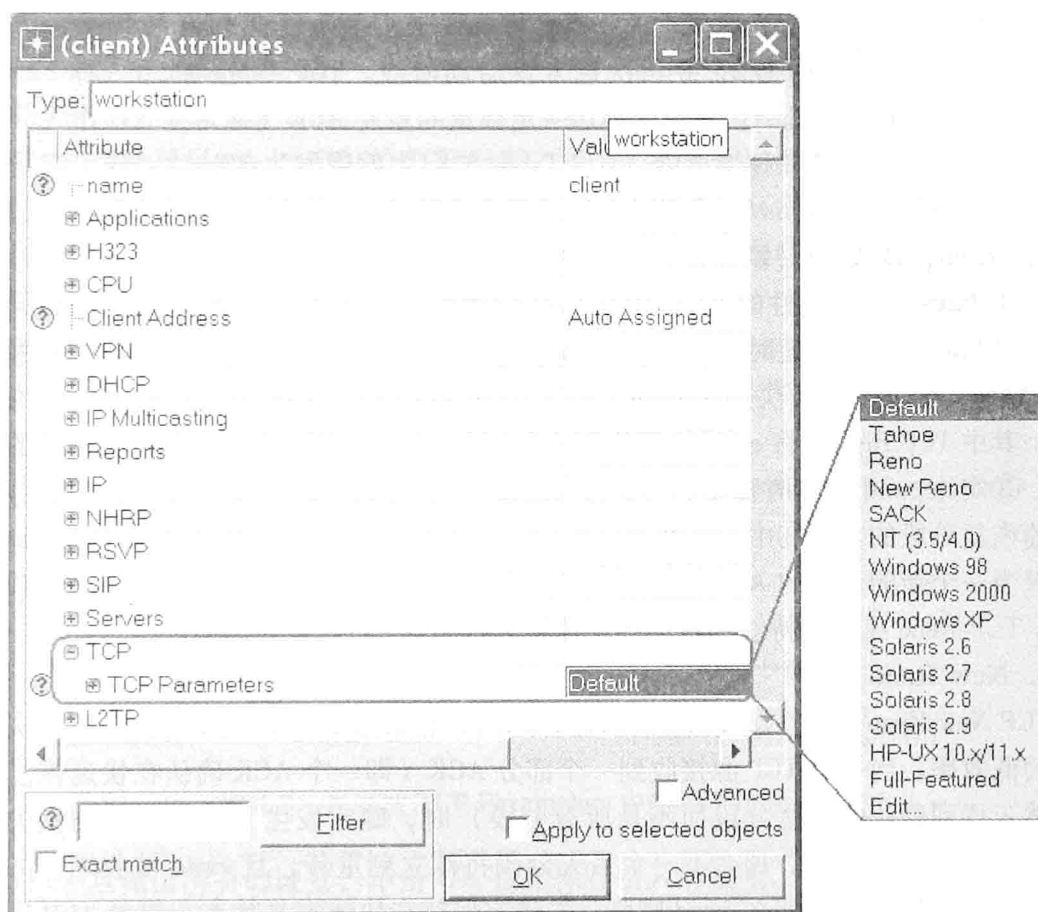


图 8.2　属性 **TCP Parameters** 及其预设值

TCP 实现的 TCP 配置。通过修改个体 TCP 配置属性，可实现这些 TCP 风格和设置的行为。但是，在实践中，首先选择 TCP 风格或设置，然后将个体配置参数调整到预期，这样做是比较高效的。

在描述 TCP 的个体配置属性之前，简短地研究一下图 8.2 所示的 **TCP Parameters** 属性的预设值含义。

1. Tahoe

这是最基本的 TCP 版本，由两个主要阶段组成：**Slow Start**（慢启动）和 **Congestion Avoidance**（拥塞避免）。在讨论这些阶段之前，回顾一下，TCP 传输速率是通过一个拥塞窗口（*cwnd*）控制的，*cwnd* 是定义一个 TCP 源可以发送但未被确认的分段数的值。最初情况下，当一条 TCP 连接开始传输数据时，它进入慢启动阶段。在慢启动过程中，拥塞窗口的尺寸指数性地增长，直到它达到慢启动拥塞阈值（*ssthresh*）。一旦发生这种情况，TCP 切换到拥塞避免阶段，其中 *cwnd* 线性增加。典型情况下，*cwnd* 的初值是一个可配置的参数，且在 OPNET 中，是通过属性 **Slow - Start Initial Count (MSS)** 指定的。慢启动拥塞阈值的初值被设置为 64KB 和目的地主机接收缓冲尺寸这两者的较小值。在 TCP 中，如果源 TCP 进程在某个时间段内没有接收到分段确认

（ACK），则发生超时。当超时发生时，TCP 将 *cwnd* 的值重置为其初值，将 *ssthresh* 设置为超时发生时 *cwnd* 值的 1/2，并再次进入慢启动阶段。TCP Tahoe 利用一种 **Fast Retransmit**（快速重传）机制，它使用到达一定数量的重复 ACK（通常为 3）作为分段丢失的一个指示。在这样一种情形中，TCP Tahoe 的行为就像发生超时时一样：它重传丢失的分段，更新 *cwnd* 和 *ssthresh* 值，并进入慢启动阶段。TCP Tahoe 不支持快速恢复。

2. Reno，Default（默认的）

TCP Parameters 属性的 Reno 和 Default 设置对应于实现 TCP Reno 风格的相同 TCP 配置。最初状态和发生超时时，TCP Reno 的行为与 TCP Tahoe 的相同。但是，不像 TCP Tahoe 的是，在发生一次快速重传时，TCP Reno 启动一个 **Fast Recovery**（快速恢复）阶段，其中 TCP Reno①将 *cwnd* 设置为其以前值的 1/2；②将 *ssthresh* 设置为 *cwnd* 的一个新值；③实施 *cwnd* 的临时膨胀和收缩；④到达一个新的 ACK 时，进入拥塞避免阶段。在快速恢复阶段的步骤③中，在每次接收到一条重复的 ACK 时，TCP Reno 将其 *cwnd* 膨胀，仅当一个新的分段的 ACK 到达时，TCP Reno 才将 *cwnd* 缩小为它在膨胀之前的值。因此，TCP Reno 风格支持快速重传和快速恢复机制。

3. New Reno

TCP New Reno 是对 TCP Reno 的改进，原因是它消除了与单个数据窗口内多次丢失有关的低效率。当一个 TCP 源接收到一个部分 ACK（即一个 ACK 确认在快速恢复开始时仍然未得到确认的一些分段而不是所有分段）时，确定发生了多个分段的丢失。如果接收到一个部分 ACK，那么下一个丢失分段将被立刻重传，且 *cwnd* 被收缩。如果接收到更多的重复 ACK，那么 *cwnd* 膨胀。发送方保持在快速恢复状态，直到 TCP 进程接收到所有数据（当快速恢复启动时还没有被确认）的 ACK。结果，当在同一窗口内发送多次丢失时，与在 Reno 发生的情况一样，*cwnd* 并不多次减小。

4. SACK

针对拥塞控制，这种风格同时使用 Reno 和 TCP 的 **Selective Acknowledgment**（**SACK**）选项。与累加性的常规 ACK 不同，一个选择性的 ACK 被用来仅确认一个特定的分段。当一个接收方确定缓冲乱序的分段并选择性地确认分段时，当超时发生时发送方可忽略这些分段的重传，且仅需要重传仍然没有被确认的那些分段。使用 SACK 选项，使 TCP 的行为更像选择性重复协议，并改进了协议效率。

TCP Parameters 属性的其他预设值配置各种架构特定的 TCP 配置，这里不做描述。可仔细检查对应于一个特定预设值的 TCP 属性设置，方法是首先将 **TCP Parameters** 属性设置为期望的值，之后展开该属性来查看实际设置。

8.3 TCP 配置属性

在本节，描述最常用的 TCP 配置属性。如在前面提到的，支持 TCP 的任何节点模型，都包含在 **TCP... TCP Parameters** 之下的所有配置属性。一般而言，为在一个端节点上配置 TCP，需要实施如下步骤：

1）打开端节点（如工作站或服务器）的 **Edit Attributes** 对话框。

2）展开复合属性 **TCP**。

3）将属性 **TCP Parameters** 设置为图 8.2 所示的预设值之一。

4）如果希望指定一个定制的 TCP 配置，那么需要展开属性 **TCP Parameters**，并为关注的属性指定值。图 8.3 给出可用的 TCP 配置属性。在位于属性左侧一个圆圈内的一个问号表示的图标上单击，打开一个 **Attribute Description** 窗口，该窗口提供相应属性的一个简短描述和含义。

Attribute	Value
TCP Parameters	(...)
Version/Flavor	Unspecified
Maximum Segment Size (byt...	Auto-Assigned
Receive Buffer (bytes)	65535
Receive Buffer Adjustment	None
Receive Buffer Usage Thre...	0.0
Delayed ACK Mechanism	Segment/Clock Based
Maximum ACK Delay (sec)	0.200
Maximum ACK Segments	2
Slow-Start Initial Count (MSS)	2
Fast Retransmit	Enabled
Duplicate ACK Threshold	3
Fast Recovery	Disabled
Window Scaling	Disabled
Selective ACK (SACK)	Disabled
ECN Capability	Disabled
Segment Send Threshold	MSS Boundary
Active Connection Threshold	Unlimited
Nagle Algorithm	Disabled
Karn's Algorithm	Enabled
Timestamp	Disabled
Initial Sequence Number	Auto Compute
Retransmission Thresholds	Attempts Based
Initial RTO (sec)	1.0
Minimum RTO (sec)	0.5
Maximum RTO (sec)	64
RTT Gain	0.125
Deviation Gain	0.25
RTT Deviation Coefficient	4.0
Timer Granularity (sec)	0.5
Persistence Timeout (sec)	1.0
Connection Information	Do Not Print

图 8.3　在属性 **TCP Parameters** 中的 TCP 配置属性

5）一旦做出所有的改变，单击 **OK** 按钮保存引入的改变。

现在，仔细研究用来指定一个定制 TCP 配置的一些基本 TCP 属性：

1）**Version/Flavor**（版本/风格）指定 TCP 风格或设置的名字。这个属性仅可通过选择 TCP Parameters 属性的预设值之一的方式进行设置。这是一个只读属性，一旦在父属性 TCP Parameters 中指定风格或设置，这个属性就不能改变。这个属性对仿真没有影响，除了如下事实，即其值确定用于其它一些 TCP 属性所用的值，这些属性接下来会影响仿真。

2）**Maximum Segment Size**（**MSS**）［最大分段尺寸（MSS）］以字节（B）为单位指定将在一个 TCP 分段内传输的最大数据量。通常情况下，MSS 的值是依据基础网络的特征设置的，从而使被传输的 TCP 分段在 IP 层不被分段。为了避免由低层产生的分段，将 MSS 设置为相应接口的最大传输单元（MTU）尺寸减去 TCP 和 IP 首部的尺寸（通常每个首部有 20B）。例如，如果接口 MTU 为 1500B，那么 MSS 应该被设置为 1460B。这个属性也接受 Auto - Assigned（自动指派的）值，这将使仿真依据节点的接口 MTU 尺寸自动地计算 MSS 值。如果节点包含多个接口，那么依据第一个接口（即 IF0）的 MTU 尺寸计算 MSS 值。

3）**Receive Buffer**（接收缓冲）以字节为单位指定缓冲的尺寸，在所接收数据被转发到较高层之前保持接收到的数据。由 TCP 通告的窗口尺寸是在这个缓冲中可用空间总量。其值可被设置为 Default，其中该属性的值被设置为至少是所协商 MSS 的 4 倍，最大值为 64KB。

4）**Delayed ACK Mechanism**（延迟的 ACK 机制）指定被延迟的“无数据”ACK 如何由 TCP 进行传输。正常情况下，各 ACK 是由以反向传输的数据分段捎带的。但是，如果在反向没有数据要发送，那么在返回一个“无数据”的 ACK 之前，TCP 将等待一会儿。这个属性接受两个值：

① Clock Based（基于时钟的）。在传输一个“无数据”ACK 之前，TCP 将等待一个配置好的秒数［由属性 **Maximum ACK Delay**（**sec**）控制］。

② Segment/Clock Based（基于分段/时钟的）。在传输一个“无数据”的 ACK 之前，TCP 将等待一个配置好的秒数或直到接收到一个指定数量的分段［由 Maximum ACK Segments（最大 ACK 分段数）控制］。

5）**Maximum ACK Delay**（**sec**）［最大 ACK 延迟（秒）］指定在发送一个“无数据”的 ACK 之前一个 TCP 过程将等待的时长（以 s 为单位）。默认情况下，这个属性被设置为 0.2s。

6）**Maximum ACK Segments**（最大 ACK 分段数）指定在发送一个“无数据”的 ACK 之前 TCP 必须接收的分段最大数量。仅当 **Delayed ACK Mechanism**（延迟的 ACK 机制）被设置为 Segment/Clock Based 时，才使用这个属性。默认情况下，这个属性被设置为 2 个分段。

7）**Slow - Start Initial Count**（**MSS**）［慢启动初始计数（MSS）］指定在慢启动阶段开始时初始 *cwnd* 值。指定这个属性为 MSS 的整数倍，默认情况下设置为 2。

8）**Fast Retransmit**（快速重传）指定 TCP 进程是否使用快速重传。如果设置为 Enabled（激活的），那么 TCP 将在接收到一定数量［通过属性 **Duplicate ACK Threshold**（重复 ACK 阈值）加以控制］的重复 ACK 时，重传一个分段。

9）**Duplicate ACK Threshold**（重复 ACK 阈值）指定触发快速重传阶段启动的重复 ACK 数量。默认情况下，这个属性被设置为 3。

10）**Fast Recovery**（快速恢复）指定在这个节点上是否激活快速恢复机制。这个属性接收如下值：Disabled（禁止）（在这个节点处不使用快速恢复）、Reno（使用快速恢复的 Reno 变种）或 New Reno（使用 New Reno 变种）。

11）**Window Scaling**（窗口缩放）指定在连接建立过程中一条 TCP SYN 报文是否激活窗口缩放选项。对于要被激活的窗口缩放（RFC 1323），客户端节点必须使其 **Window Scaling** 属性设置为 Enabled，且服务器必须使 **Window Scaling** 设置为 Enabled 或 Passive（被动的）。

12）**Selective ACK**（**SACK**）［选择性 ACK（SACK）］确定是否使用选择性 ACK（RFC 2018）。如果要用 SACK 选择，则在客户端主机中必须将这个属性设置为 Enabled，将服务器主机中的这个属性设置为 Enabled 或 Passive。与直接设置这个属性值的做法相反，这里建议将 **TCP Parameters** 属性设置为 SACK。

13）**ECN Capability**（ECN 能力）指定 TCP 是否配置为支持 **Explicit Congestion Notification**（显式拥塞通知）（RFC 3168）。如果客户端节点将这个属性设置为 Enabled，那么它将通告它是支持 ECN 的。仅当服务器节点将 ECN Capability（ECN 能力）

设置为 Enabled 或 Passive，且它从客户端节点接收到一条 ECN 通告时，服务器节点才通告对 ECN 的支持。当客户端和服务器都同意使用 ECN 时，在这条连接上的所有报文才将其首部进行设置，指明这样的事实。正常情况下，经历拥塞的路由器将丢弃通过它的报文。但当一条支持 ECN 的报文通过正经历拥塞的一台路由器时，路由器将不会丢弃该报文，但相反将在报文中的首部比特做标记，指明路由器遇到拥塞。报文的接收方将拥塞指示回传给发送方，发送方必须通过实施拥塞控制来做出响应，就像报文被丢弃了一样。

14）**Active Connection Threshold**（活跃连接阈值）指定在节点上所允许的活跃 TCP 连接的最大数量。

15）**Nagle Algorithm**（Nagle 算法）指定是否激活 Nagle 算法（RFC 1122）。如果存在以前所发送数据的待发（outstanding）ACK，则 Nagle 算法防止比一个 MSS 小的分段进行传输。这个属性接受如下值：①Disabled，在这个节点上不使用 Nagle 算法；②Enabled，在这个节点中使用 Nagle 算法；③Per - Send（每次发送时使用），实现这个算法的一个优化版本，其中对待发 ACK 的检查，是在每发送基础上而不是每分段基础上进行的。这里，“发送”指来自较高层应用的单个发送操作。

16）**Karn Algorithm**（Karn 算法）指定计算重传超时（RTO）值的 Karn 算法（RFC 2988）是否在这个节点上使用。这个属性仅接受两个可能值：Disabled 和 Enabled。当更新用于 RTO 计算的往返时间估计时，Karn 算法忽略被重传的分段。往返时间估计仅基于无二义性的 ACK，这些 ACK 是针对仅被发送一次的分段发出的。

17）**Timestamp**（时间戳）指明是否支持 TCP 时间戳选项（RFC 1323）。**Timestamp** 是一个复合属性，包括子属性 **Status**（状态）（指定是否激活该选项）和 **Clock Tick**（**millisec**）[时钟滴答（毫秒）]（指明用来增加时间戳值的数值）。要激活这个选项，客户端节点必须将其 Status 属性设置为 Enabled，且服务器节点必须将这个属性设置为 Enabled 或 Passive。

18）**Initial Sequence Number**（初始序列号）指定在这个节点处所有连接上外发分段所用的初始序列号。这个属性接受值 Auto Compute（自动计算），其中为每条连接随机选择初始序列号。

19）**Initial RTO**（**sec**）、**Minimum RTO**（**sec**）和 **Maximum RTO**（**sec**）[初始 RTO（秒）、最小 RTO（秒）和最大 RTO（秒）] 遵循 Karn 算法，在一条连接的生命期上为 RTO 指定初始、最小和最大值。默认情况下，这些值分别被设置为 3s、1s 和 64s。

其他 TCP 配置属性描述各种高级的 TCP 功能特征，本书中不做讨论。欲了解有关 OPNET TCP 实现的更多信息，参见用户指南，通过 **Project Editor** 下拉菜单中的 **Protocols→TCP→Model User Guide**（协议→TCP→模型用户指南）选项可进行访问。另外，OPNET 软件包括称为 **TCP** 的一个标准范例项目。这个项目包含几个场景，提供了各种 TCP 功能特征的卓越展示，并可作为评估 TCP 性能如何开发仿真场景的绝佳参考。

8.4 常用传输协议统计量

传输层统计量在全局统计量和节点统计量中都有。全局统计量包含称为 TCP 的单个类，监测所有节点上 TCP 的性能。没有 UDP 的全局统计量。但是，节点统计量包含评估个体节点内传输层性能的三个类：**TCP**、**TCP Connection** 和 **UDP**。节点统计量的 **TCP** 和 **UDP** 类分别收集在每个节点处所有 TCP 和 UDP 流量的数据。**TCP Connection** 类为一个节点处的每条 TCP 独立地收集并维护统计信息。表 8.1 给出所有可用传输层统计量的一个简短描述。

表 8.1　传输层统计量的汇总

类	名字	描述
Global Statistics TCP（全局统计量 TCP）	*Delay*（*sec*）[延迟（秒）]	这个统计量以秒为单位记录所有 TCP 报文经历的平均网络范围的延迟。是一条应用层报文从源 TCP 层发出直到完全由目的地节点的 TCP 层接收的时间测量得到的
	Retransmission Count（重传计数）	这个统计量记录网络中 TCP 分段重传的总数
	Segment Delay（*sec*）[分段延迟（秒）]	这个统计量以秒为单位记录所有 TCP 分段的平均网络范围的延迟。是一个 TCP 分段从源 TCP 层发出直到分段被目的地节点中的 TCP 层接收到的时间测量得到的。这不同于 Delay（packet delay）[延迟（报文延迟）]，原因是如果报文尺寸大于 MSS，那么一条应用层报文可作为多个分段由 TCP 传输
Node Statistics TCP（节点统计量 TCP）	*Active Connection Count*（活跃的连接计数）	这个统计量记录在这个节点处活跃的 TCP 连接总数
	Connection Aborts（连接中止数）	这个统计量记录在这个节点处由高层中止的 TCP 连接总数
	Delay（*sec*）[延迟（秒）] *Retransmission Count*（重传计数） *Segment Delay*（*sec*）[分段延迟（秒）]	这些统计量与相应的全局统计量的含义相同，例外是不像全局统计量在系统中的所有节点上聚集结果，这些统计量是独立地针对每个节点记录的

（续）

类	名字	描述
Node Statistics TCP（节点统计量 TCP）	*Load*（*bytes*）[负载（字节）] *Load*（*bytes/sec*）[负载（字节/秒）] *Load*（*packets*）[负载（报文数）] *Load*（*packets/sec*）[负载（报文数/秒）]	这些统计量记录在这个节点处由应用层提交到 TCP 层的数据总量或速率。它包括该节点处所有 TCP 连接的数据。这些统计量的单位是字节数、每秒字节数、报文数和每秒报文数
	Traffic Received（*bytes*）[接收到的流量（字节）] *Traffic Received*（*bytes/sec*）[接收到的流量（字节数/秒）] *Traffic Received*（*packets*）[接收到的流量（报文数）] *Traffic Received*（*packets /sec*）[接收到的流量（报文数/秒）]	这些统计量记录在这个节点处由 TCP 层接收并转发到应用层的数据总量或速率，包括该节点处所有 TCP 连接的数据。这些统计量的单位是字节数、每秒字节数、报文数和每秒报文数
Node Statistics TCP Connections（节点统计量 TCP 连接）	*Congestion Window Size*（*bytes*）[拥塞窗口尺寸（字节）]	这个统计量为每条 TCP 连接记录拥塞窗口的尺寸
	Delay（*sec*）[延迟（秒）] *Retransmission Count*（重传计数） *Segment Delay*（*sec*）[分段延迟（秒）] *Load*（*bytes*）[负载（字节）] *Load*（*bytes/sec*）[负载（字节/秒）] *Load*（*packets*）[负载（报文数）] *Load*（*packets/sec*）[负载（报文数/秒）] *Traffic Received*（*bytes*）[接收到的流量（字节）] *Traffic Received*（*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received*（*packets*）[接收到的流量（报文数）] *Traffic Received*（*packets /sec*）[接收到的流量（报文数/秒）]	这些统计量与相应的节点统计量的有相同的含义，例外是，不像节点统计量是在每个节点基础上收集结果的，这些统计量是针对每条 TCP 连接独立地进行记录的
	Flight Size（*bytes*）[传输尺寸（字节）]	这个统计量记录每条连接的待确认（即在传输中）的数据量。这个统计量是每次一条 TCP 连接传输一个分段或接收到一个 ACK 时进行更新的

（续）

类	名字	描述
Node Statistics TCP Connections（节点统计量 TCP 连接）	*Received Segment ACK Number*［接收到的分段 ACK 号］ *Received Segment*（接收到的分段） *Sequence Number*（序列号）	这些统计量相应地为一条连接记录接收到的 ACK 和序列号
	Remote Receive Window Size（*bytes*）［远端接收窗口尺寸（字节）］	这个统计量以字节为单位记录远端节点处为一条连接通告的接收窗口的尺寸
	Retransmission Timeout（*sec*）［重传超时（秒）］	这个统计量为一条连接记录 RTO 值
	Segment Round - Trip Time（*sec*）［分段往返时间（秒）］ *Segment Round - Trip Time Deviation*（分段往返时间偏差）	这些统计量以秒为单位记录一条连接的分段往返时间的均值，同时记录其偏差
	Selectively ACKed Data（*bytes*）［选择性确认的数据（字节）］	这个统计量记录由一条连接发送的并使用 SACK 选项确认的字节总数。没有使用 SACK 机制而确认的数据不会由这个统计量记录
	Send Delay（*CWND*）（*sec*）［发送延迟（CWND）（秒）］ *Send Delay*（*Nagle' s*）（*sec*）［发送延迟（Nagle 的）（秒）］ *Send Delay*（*RCV - WND*）（*sec*）［发送延迟（RCV - WND）（秒）］	这个统计量以秒为单位记录在发送缓冲数据中遇到的延迟。对延迟的三种不同原因，都单独地有这个统计量：CWND、Nagle 的和 RCV - WND。延迟值是从 TCP 开始约束发送数据（由于特定原因）的时间知道数据被发送出去的时间测量得到的。这三种原因是：CWND 是由于拥塞窗口太小，Nagle 的是当延迟由 Nagle 算法导致的，而 RCV - WND 是由于接收窗口太小
	Sent Segment ACK Number（发送分段 ACK 号） *Sent Segment Sequence Number*（发送分段序列号）	这些统计量分别记录一条连接的发送 ACK 和分段号

（续）

类	名字	描述
Node Statistics UDP （节点统计量 UDP）	*Traffic Received* （*bytes/sec*）[接收到的流量（字节/秒）] *Traffic Received* （*packets/sec*）[接收到的流量（报文数/秒）]	这些统计量记录由 UDP 从低层接收到并转发到高层的数据总量或速率
	Traffic Sent （*bytes/sec*）[发送的流量（字节/秒）] *Traffic Sent* （*packets/sec*）[发送的流量（报文数/秒）]	这些统计量记录由 UDP 从高层接收到要传输的数据总量或速率

第 9 章　网络层——IP 介绍

9.1　引言

在如今的网络中，存在几种不同的层 3（Layer 3）或网络层协议。但是，IP 是最广泛使用的网络层协议。IP 是一个十分复杂的协议，经常被看作将如今的互联网黏合在一起的黏合剂。即使 IP 具有广泛的职责，但其主要任务是在互联网的端系统之间交付数据。这种数据交付可跨越多条链路和多个网络，并要求每个节点通过一个 IP 地址是唯一标志的。一个 **IP address**（IP 地址）标志一个特定接口，该接口将一个节点连接到一个网络。在端系统或主机中，典型地包含单个接口，一个 IP 地址唯一地标志节点的接口或节点本身，因此经常称为主机的 IP 地址。另外，路由器典型地被连接到多个网络，因此包含多个接口。一台路由器的每个活跃接口都有分配给它的一个不同 IP 地址。在本书中，将一个 IP 地址称为识别一个节点中一个特定接口的地址。除了端到端通信外，网络层还处理数据封装、分片和组装、数据转发、路由、压缩、组播、负载均衡、支持各种服务质量（QoS）需求的首选的数据处理（preferential data treatment）以及其他功能。OPNET 软件对这些功能特征的绝大多数都进行了建模，并允许 IP 各方面的定制配置。

通过为较快速和较容易地配置各种被仿真系统的功能特征而添加新的选项，OPNET 技术公司一直持续地在改进其软件 GUI。作为这种增长的结果，通过实施配置步骤的不同集合，会经常得到相同的系统配置。对于 IP 而言，情况特别真实，自最早的发行版本以来，IP 就得到 OPNET 软件的支持。特别地，OPNET 软件允许指定各种 IP 功能特征，采用诸如下面的配置工具：

1）在个体节点内的模型属性和配置参数。

2）与整体仿真有关的全局属性和配置参数。

3）在 **Project Editor** 中的 **Protocols** 下拉菜单选项。

4）工具节点（用来定义各种场景范围的功能特征的对象），使所定义的概要或机制可由场景内的任意对象访问。

在第 9 章和第 10 章，描述 OPNET 支持的各种 IP 功能特征的配置步骤。描述配置 IP 功能特征的最普遍和最容易的方法，这样会偶尔忽略掉某些配置替代方法，这些方法是太过烦人的和容易出错的。我们并不意图提供 OPNET 中所有可用 IP 功能特征和所有配置它们的可能方法的全面深入描述。但是，这两章提供了由 OPNET 支持的主要 IP 功能特征的详细描述。具体而言，本章开始时在 9.2 节引入端节点和核心节点模型中可用的基本 IP 配置属性。接下来在 9.3 节继续描述 IP 寻址和在节点接口上指定 IP 地址和

子网掩码的步骤。在9.4节提供了OPNET所支持的IP功能特征（如压缩方案、路由、接口和负载均衡）配置步骤的一个概述。9.5节提供了互联网控制消息协议（ICMP）的描述及其在OPNET中所支持的属性。本章在9.6节收尾，给出IP网络性能评估的常见IP统计量和其他功能特征。在第10章讨论高级IP功能特征，如网络地址转换（NAT）、IP组播、IP版本6（IPv6）和QoS。

9.2 基本IP配置属性

在OPNET中，支持层3功能的每个节点模型（如路由器、网关、服务器、工作站、云、LAN和其他对象）都包含一个**IP**模块，实现IP的各种功能特征。注意集线器和层2交换机的节点模型不包含IP模块，因此在本章中不做讨论。典型情况下，配置IP的模型属性聚集在称为**IP**的单个复合属性下。但是，配置IP的子属性的数量和类型取决于节点模型的类型。例如，端节点（如工作站和服务器）通常并不要求所有的IP功能。由此，相比于核心节点模型（如路由器、网关、层3交换机和IP云），它们包含较少的可由用户配置的IP配置属性，典型情况下，核心节点模型将包含大量的IP属性。但是，依据一个节点的具体职能，属性的数量也是变化的。因为IP云的配置变得十分复杂，隐藏了不太可能需要的IP属性，以此避免集群（clutter），并使IP配置更加具备可管理能力。本节描述存在于端节点和核心节点模型中的主要IP配置属性。略去了一些比较高级属性的描述，这些属性仅存在于特定节点模型内。

9.2.1 一个端节点模型的基本IP配置属性

为了在一个端节点模型中配置IP属性，需要打开**Attributes**窗口，方法是右击一个节点并选择**Edit Attributes**选项。除了使用互联网组管理协议（IGMP）配置IP组播的属性之外，指定IP性质的所有属性都被安排在称为**IP**的复合属性下，由三个复合子属性组成：

1）**IP Host Parameters**（IP主机参数）指定在这个节点上的IP寻址和转发功能特征。

2）**IP Processing Information**（IP处理信息）定义如下硬件特征，如处理器或槽的数量、交换和转发速率以及可用于报文转发的主存容量。

3）**IP QoS Parameters**（IP QoS参数）定义在节点处部署的QoS机制。将在第10章讨论QoS配置。

IP Host Parameters由复合属性**Interface Information**（接口信息）和在节点处配置各种路由性质的几个其他属性组成。属性**Interface Information**指定节点的接口特点，并由如下子属性组成：

1）**Name**，即接口的一个由字母数字组成的名字。在OPNET中，一个节点中的每个接口有一个名字，以字母IF（表示接口）开始，后跟接口号。OPNET以从0开始的升序对节点接口分配号码。因为每个节点仅有一个接口，所以一个端节点中这个属性的

值为 IF0，如图 9.1 所示。

图 9.1 一个端节点中的 IP 模型属性

2）**Address** 指定接口的 IP 地址。

3）**Subnet Mask**（子网掩码）指定接口的子网掩码。在 9.3 节相当详细地讨论属性 **Address** 和 **Subnet Mask**。

4）**MTU（bytes）**［MTU（字节）］是最大传输单元（MTU）的值，是基础网络可接受的最大可能报文尺寸。MTU 是这样的参数之一，它确定在将进来的数据转发到网络之前 IP 如何对之实施分片。OPNET 提供几个预设 MTU 值：Ethernet（以太网），1500B；FDDI，4470B；ATM，4470B；IP，1500B；Token Ring（令牌环），4464B；Frame Relay（帧中继），1500B；WLAN，2304B；或 TDMA，2304B。MTU 最小允许值为 20B，对应于 IP 首部的尺寸。

5）**Compression Information**（压缩信息）是这样一个属性，它指定在这个接口上使用的压缩方案的名字。通过工具对象 **IP Attribute Config**（IP 属性配置）定义实际方案，将在 9.4.1 节讨论这个工具对象。

6）**IPv6 Parameters**（IPv6 参数）为 IPv6 指定地址和其他配置参数。

7）**Description**（描述）是这个接口的一个文本描述。这个属性对仿真没有影响。

8）**Layer 2 Mappings**（层 2 映射）指定这个接口和层 2 永久虚拟电路（PVC）［如

ATM、帧中继（FR）或虚拟局域网（VLAN）连接］之间的映射。

IP Host Parameters 的其他子属性描述如下：

1）**Passive RIP Routing**（被动 RIP 路由）指定当前端节点是否依据从路由信息协议（RIP）得到的路由信息而转发数据，RIP 运行在被动模式。

2）**Default Route**（默认路由）指定默认网关的 IP 地址。如果 IP 路由被关闭或如果 IP 转发表没有产生到目的地的路由，那么节点将报文转发到默认网关。属性 **Default Route** 的默认设置是 Auto Assigned（自动指派的），这意味着在仿真过程中默认网关地址将被自动地确定。

3）**Static Routing Table**（静态路由表）定义 IP 版本 4（IPv4）静态路由表。静态路由表中的每个表项由属性组成，如目的地的 IP 地址和掩码、在到目的地路径上下一跳的 IP 地址以及指定路由优先级的管理权重。如果到一个目的地有多条路由，那么将选择具有最小管理权重的路由。如果没有激活路由协议或如果静态路由表中的一条路由具有比一个路由协议（如 RIP）提供的一条路由较低的管理权重，那么该节点使用静态路由表。

4）**IPv6 Default Route**（IPv6 默认路由）指定一个默认网关的 IPv6 地址。

5）**Multicast Mode**（组播模式）确定当前节点是否支持组播应用。这个属性仅接受两个可能值：Enabled 和 Disabled。

6）**V6 Static Routing Table**（V6 静态路由表）指定 IPv6 静态路包括表。这张表中每个表项包括目的地 IPv6 地址、前缀长度（在地址中最左面比特数量，定义地址的网络部分）、下一跳的 IPv6 地址和管理权重。

9.2.2　一个核心节点的基本 IP 配置属性

如图 9.2 所示，核心节点模型也在复合属性 **IP** 下有 IP 配置参数，包括如下子属性：

1）**APS Parameters**（APS 参数）为在网络中支持容错，配置自动保护切换（APS）机制。APS 配置在主接口失效事件下的备份接口。这是各种 Cisco 路由器中存在的一个高级功能特征，在本书中不做进一步讨论。

2）**IP Processing Information**（IP 处理信息）指定节点的硬件特征，如处理器或槽的数量、交换和转发速率，以及可用于报文转发的主存储量。

3）**IP QoS Parameters**（IP QoS 参数）指定路由器接口上的 QoS 设置。

4）**IP Routing Parameters**（IP 路由参数）为这个节点中的每个接口指定 IP 寻址和路由协议设置。

5）**IP Slot Information**（IP 槽信息）。如果当前节点支持一个多处理器环境［即当 **IP Processing Information... Processing Scheme**（IP 处理信息... 处理方案）属性的值被设置为 Slot Based Processing（基于槽的处理）］时，配置个体处理器或槽。这个属性允许指定信息，如每个槽的服务速率以及输入和输出缓冲容量。

6）**IPv6 Parameters**（IPv6 参数）在这个节点上配置各种 IPv6 寻址和路由功能

特征。

7）**NAT Parameters**（NAT 参数）指定在这个节点上的网络地址转换（NAT）配置。

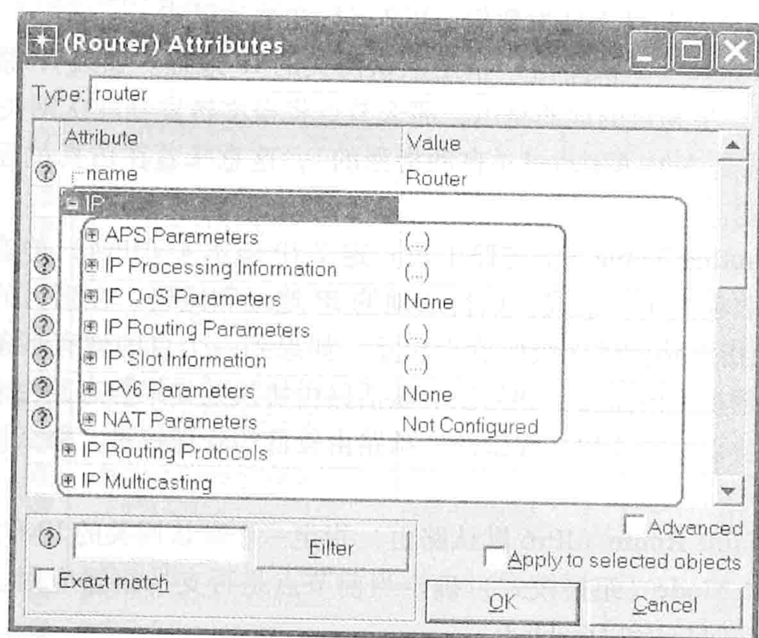


图 9.2 一个核心节点中的 IP 模型属性

IP Routing Parameters 复合属性与端节点模型中的 **IP Host Parameters** 属性的目的相同。但是，如图 9.3 所示，通过在端节点中不存在的各种附加子属性，这个属性支持配置广泛的附加功能特征。下面提供了最常用的 **IP Routing Parameters** 子属性的一个汇总：

1）**Router ID**（路由器 ID）指定由开放最短路径优先（OSPF）路由协议使用的一个唯一路由器标识符。

2）**Autonomous System Number**（自治系统号）是一个 16 位的数字，表示 1 和 65 535之间的一个整数值，唯一地标志这个路由器所属的一个自治系统（AS）。典型情况下，AS 号为边界网关协议（BGP）所用。

3）**Interface Information**（接口信息）是一个复合属性，为这个节点的每个物理接口指定编址和路由信息。

4）**Aggregate Interfaces**（聚合接口）是配置链路聚合控制协议（LACP）的一个复合属性，方法是将多个物理接口映射到单个逻辑或聚合接口。这个属性定义聚合接口的特征。将一个物理接口映射到这个聚合接口，要求将属性 **IP... IP Routing Parameters... Interface Information... < interface name >... Aggregation Parameters... Aggregate Interface**（IP... IP 路由参数... 接口信息... <接口名>... 聚合参数... 聚合接口）的值设置为这个聚合接口的名字。在 9.4.3 节讨论配置聚合接口的步骤。

5）**Loopback Interfaces**（环回接口）是一个复合属性，用于在这个节点上配置环

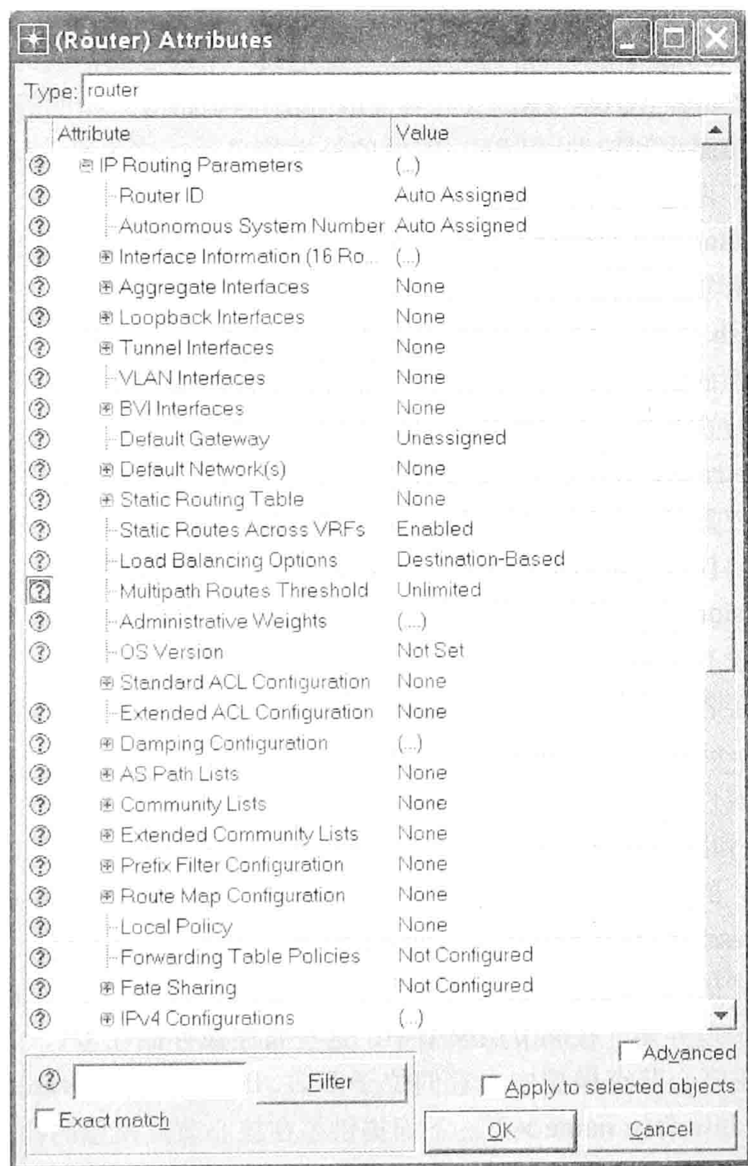


图 9.3　**IP Routing Parameters** 复合属性

回接口。默认情况下，所有路由器模型包括单个环回接口。通过配置 **Loopback Interfaces** 属性，可添加附加的环回接口。

6）**Tunnel Interfaces**（隧道接口）是在这个节点上定义隧道接口的一个复合属性。

7）**VLAN Interfaces**（VLAN 接口）是配置 VLAN 接口的一个复合属性。只能在包括一个层 3 卡的交换节点模型（如 *ethernet16_ switch_ with_ layer3*）中指定这 8 个属性。在其他节点模型中，这作为具有值 None 的一个简单属性出现。

8）**BVI Interfaces**（BVI 接口）是指定桥接 VLAN 接口配置参数的一个复合属性。

9）**Default Gateway**（默认网关）指定默认网关的 IP 地址。但是，这个属性已经废弃不用，且对仿真没有影响。

10）**Default Network（s）**（默认网络）是一个复合属性，允许指定默认网关。

11）**Static Routing Table**（静态路由表）是定义一个静态路由表的一个复合属性。

12）**Static Routes Across VRFs**（在各 VRF 间的静态路由）指明静态路由是否可添加到属于不同虚拟路由和转发（VRF）实例的外发接口上。这个属性仅适用于 Cisco 路由器，在本书中不做进一步的讨论。

13）**Load Balancing Options**（负载均衡选项）指定负载均衡机制的类型。在 9.4.4 节讨论负载均衡配置。

14）**Multipath Routes Threshold**（多路径路由阈值）是这样一个属性，指定在路由过程中将被考虑的到达相同目的地的路由最大数量。注意：将记录到目的地的所有可用路由，但仅有被这个属性指定的数量才被考虑用于路由。

15）**Administrative Weights**（管理权重）是指定各种路由协议优先级的一个复合属性。低的管理权重指示较高的优先级。如果 IP 转发表包含由各种协议提供的多条路由，那么由路由协议计算的具有最小管理权重值的路由，将被首先考虑采用。

16）**OS Version**（OS 版本）是指定在这个节点上所用操作系统版本的属性。但是，这个属性对仿真没有影响。

其他属性配置各种高级功能特征，如访问控制列表、路由映射、前缀过滤器和路由抑制（route dampening），在本书中不做讨论。

如上所述，属性 **Interface Information** 包含在这个核心节点模型中所有可用物理接口上配置 IP 寻址和路由特征的参数。图 9.4 给出单个物理接口（如接口 IF0）的 **Interface Information** 的子属性。注意，属性 **Name**、**Address**、**Subnet Mask**、**MTU (bytes)**、**Compression Information**、**Description** 和 **Layer 2 Mapping** 可用于核心节点模型和端节点模型中的接口配置。在这两种情形中，这些属性具有相同的含义，所以在本节不再描述。唯一差异是，在端节点模型中，这些属性被存储在 **IP... IP Host Parameters** 属性下，而在核心节点模型中，它们位于 **IP... IP Routing Parameters... Interface Information... <interface name>** 下。下面提供了在核心节点模型的物理接口上配置 IP 性质的几个其他常用属性的简短描述，同时略去配置各种高级功能特征［如 **Packet Filter**（报文过滤器）、**Policy Routing**（策略路由）、**Routing Instance**（路由实例）和 **VRF Sitemap**（VRF 站点图）］的属性描述。

1）**Status** 属性指定接口的当前管理状态。这个属性接受两个值：Active（意味着该接口配置为可运行的）和 Shutdown（关闭，意味着该接口是被禁用的，不会转发或接收任何数据）。

2）**Operational Status**（运行状态）属性指定当前接口是否是物理上运行的。仅当属性 **Status** 被设置为 Active 时，这个属性的值才是相关的。这个属性接受值 Up（如可运行的）、Down（如不可运行的）和 Infer（推断的）（例如，目前状态是未知的，且将在仿真过程中进行确定）。

3）**Secondary Address Information**（附属地址信息）允许指定附加地址，为的是在单个物理网络上支持多个逻辑子网。

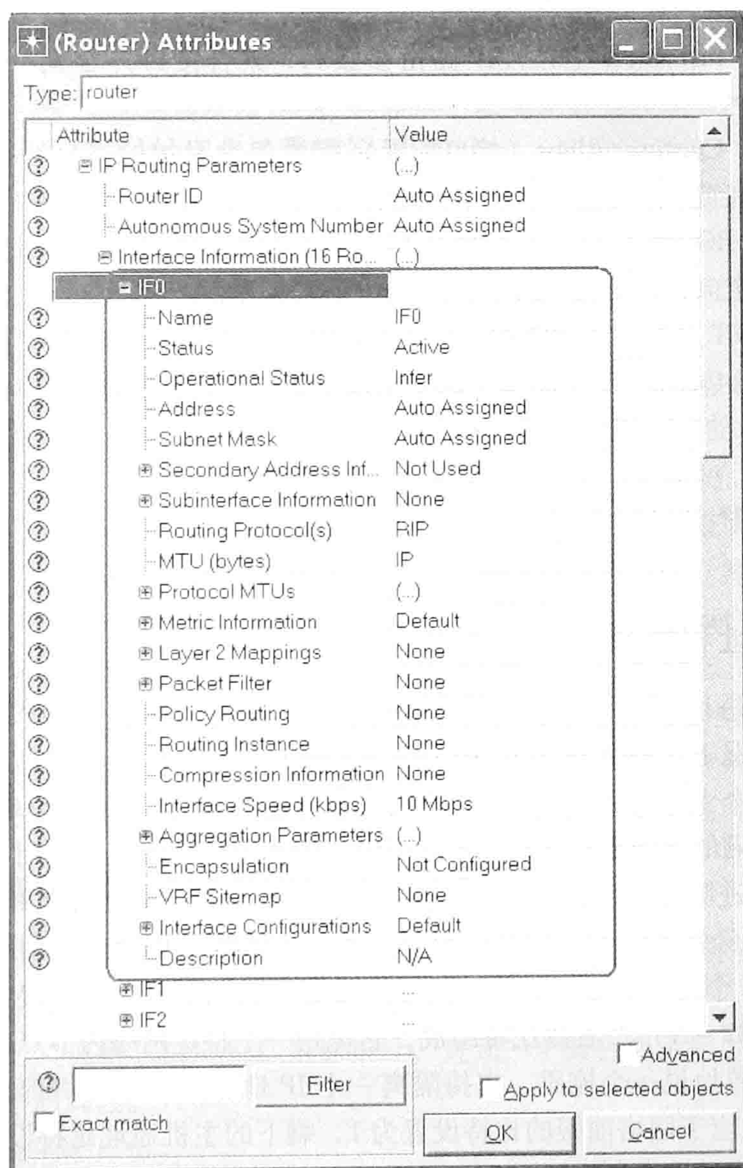


图 9.4　单个物理接口（IF0）的 **Interface Informations** 的子属性

4）**Subinterface Information**（子接口信息）指定子接口配置参数。

5）**Routing Protocol**（s）（路由协议）指定在这个接口上激活的动态路由协议。在相同接口上可激活一个以上的协议。同样，相同节点的不同接口可激活不同的路由协议。

6）**Protocol MTUs**（协议 MTU）为各种协议指定 MTU 尺寸（如 IP、MPLS 和 IPv6）。

7）**Metric Information**（度量信息）指定由路由协议使用的各种度量值（如带宽、延迟、可靠性和负载），以便确定接口开销。

8）**Aggregation Parameters**（聚合参数）指定当前物理接口被配置为逻辑聚合接口

的组成部分时的接口聚合参数。仅当通过属性 **IP... IP Routing Parameters... Aggregate Interfaces**（IP... IP 路由参数... 聚合接口）定义一个聚合接口时，才可配置这个属性。

9）**Interface Configurations**（接口配置）配置各种附加的接口特征，如指定在一个接口上是否激活定向广播，定义用于转发 UDP 广播消息［如由动态主机配置协议（DHCP）产生的 BOOTP］的一个帮助者（helper）地址，列出在接口上激活的 ICMP 特征（如 ICMP 重定向、ICMP 掩码请求和 ICMP 不可达消息）等。

请参见 OPNET 文档和属性描述，以了解存在于各种节点模型中 IP 配置属性的完整描述。可通过 **Protocols→IP→Model User Guide**（协议→IP→模型用户指南）下拉菜单选项访问 OPNET 的 IP 文档。

9.3 管理 IP 地址

9.3.1 IP 地址和掩码

当前，IPv4 是 IP 的主要版本。每个 IPv4 地址，或简单地说一个 IP 地址，是一个 32 位的值。典型情况下，一个 IP 地址是以一个点分十进制表示显示的，其中地址值的每个字节被写为一个十进制数，以一个句号分隔。因为 1 个字节对应于 8 位，所以 IP 地址的每个点号分隔的字节表示 0 和 255 之间的一个整数值。例如，使用点分十进制表示法，IP 地址的二进制值 10101000 00000000 11101010 11111111 被写作 168. 0. 234. 255。

一个 IP 地址经常被认为由两部分组成：网络前缀和主机地址。网络前缀或网络地址是 IP 地址的 *N* 个连续最左侧比特。网络前缀唯一地识别节点所连接到的网络。主机地址由剩下的（**32 - *N***）个右侧比特组成，它识别一个特定网络内节点的接口。一个 IP 地址掩码，或简单地说一个掩码，支持隔离一个 IP 地址的网络前缀部分。掩码也是一个 32 比特值，对应于网络前缀的比特设置为 1，剩下的主机地址比特设置为 0。典型情况下，掩码使用点分十进制表示或无类域间路由（CIDR）表示法进行表示。CIDR 表示法将一个掩码表示为一个斜杠（“/”）后跟网络前缀的长度。在 CIDR 表示法中，掩码被附加到一个 IP 地址的结束后面。例如，考虑以点分十进制表示法写出的 IP 地址 133. 234. 7. 56 及其掩码 255. 192. 0. 0。这个地址—掩码对以 CIDR 表示法表示的相应值是 133. 234. 7. 56/10。通过实施 IP 地址及其掩码之间的逐位 AND 运算，可容易地计算网络前缀。掩码是许多路由协议的本质元素，原因是典型情况下，路由信息是被存储的，且转发决策通常是依据一个 IP 地址的网络前缀部分做出的。

为了在一个端节点中配置 IP 地址和掩码，需要改变称为 **Address** 和 **Subnet Mask** 的属性的值，这两者都位于复合属性 **IP... IP Host Parameters... Interface Information** 下。在核心节点模型中，**Address** 和 **Subnet Mask** 属性位于复合属性 **IP... IP Routing Parameters... Interface Information... < interface number >** 下，其中 **< interface number >** 标志节点的接口之一。回顾一下，OPNET 是从 0 开始编号节点接口的，且每个接

口名以字母 IF 开头。因此，如果一个核心节点包含 n 个接口，那么接口号的范围将是从 IF0 到 IF <$n-1$>。

默认情况下，IP 地址被设置为 Auto Assigned，这意味着 **Address** 属性值将在仿真开始时被自动指派。属性 Address 也接受值 No IP Address，这意味着这个节点在仿真过程中将不是任何 IPv4 路由/转发活动的组成部分（即一个节点可被配置运行 IPv6）。在属性的 **Value** 字段上双击或选择选项 Edit...，可为节点的接口指派一个期望的 IP 地址。IP 地址值应该使用点分十进制表示法指定。指定一个无效的 IP 地址值会导致仿真不能正确地执行。

类似地，**Subnet Mask** 值也应该使用点分十进制表示法进行指定。默认情况下，**Subnet Mask** 也设置为 Auto Assigned。另外，**Subnet Mask** 属性也接受如下预定义值：

1）Class A（natural）[A 类（自然的）]，对应于 255.0.0.0 掩码。

2）Class B（natural）[B 类（自然的）]，对应于 255.255.0.0 掩码。

3）Class C（natural）[C 类（自然的）]，对应于 255.255.255.0 掩码。

注意在核心节点中，每个物理接口有其自己的 **Address** 和 **Subnet Mask** 属性集合。另外，核心节点不必使它们的所有接口在一个仿真中都处于活跃状态。因此，经常需要识别（即发现名字）将配置的接口。典型情况下，将希望仅配置那些少量活跃接口中的一些接口，这些接口对应于一个网络中将当前节点连接到其他节点的链路。注意，在边缘节点模型中这不是一个问题，该模型通常包含名为 IF0 的单个接口。

9.3.2　识别附接到一条链路的接口的名字

为了识别附接到一条链路（在一个被仿真网络中将当前节点连接到另一个对象）的一个接口的名字，需要实施如下动作之一：

1）在链路上“逗留”（Hover）鼠标指针，直到一个黄色提示框出现。

2）检查链路的属性。

3）检查链路的端口值。

如图 9.5a 所示，提示框提供有关该链路的基本信息，包括通过那条链路连接的节点的接口名。OPNET 依赖于如下接口命名惯例：*<node name>.<interface type>_<interface number>_<channel number>*（<节点名字>.<接口类型>_<接口号>_<信道号>）。为了识别接口名，仅关注于紧跟 *<interface type>* 的号码。例如，如图 9.5a 所示，该链路连接到名为 Router 的节点中接口号 10 和名为 Server 的节点中接口号 0。通过检查所连接链路的发送器和/或接收器，可得到同样的信息，如图 9.5b 所示。最后，通过右击也可识别接口名并选择 **Edit Ports** 选项，打开一个 **Select Port Assignment**（选择端口指派）窗口。如图 9.5c 所示，**Select Port Assignment** 窗口列出附接于这条链路的节点接口的名字。因此，如果希望配置 Router 使用的接口，连接 Server，那么将需要改变名为 IF10 的接口的属性值。注意，正如所预期，作为一个端节点的 Server，具有名为 IF0 的单个接口。

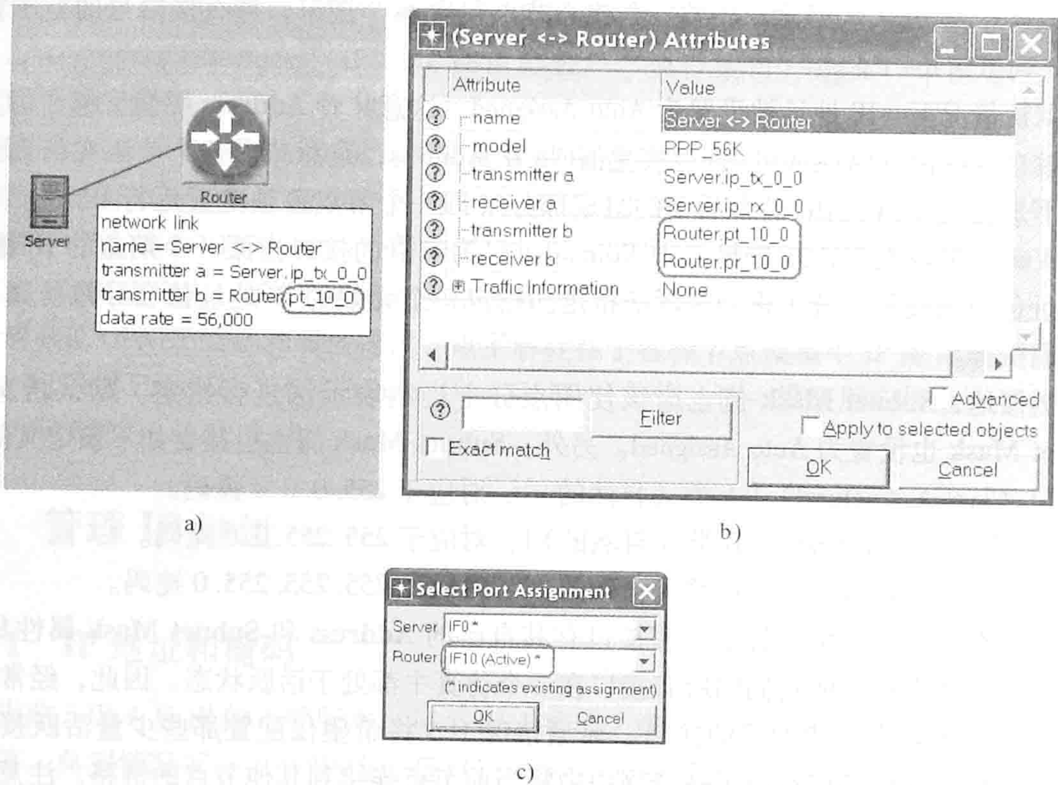


图 9.5 识别一条链路之端点的接口数的三种方法

a）在链路之上“悬停”鼠标指针之后出现黄色提示框 b）帮助识别端口号的链路属性 c）通过 Edit Ports（编辑端口）选项识别接口名

9.3.3 IP 地址配置中的常见错误

在配置 IP 地址中常见的初级错误之一是，在两个节点之间的一条链路所附接的接口上指定不同的 *network*（网络）地址（如通过一个 IP 地址或掩码属性）。正确的配置要求附接到同一条链路的 IP 地址和接口的掩码解析到相同的网络地址。

考虑如图 9.5 所示的例子和在表 9.1 中给出的不正确接口配置的两种可能场景。在场景 1 中，由于 IP 地址指派中的一个错误，连接链路在相应的节点接口上指定了不同的网络地址。Server 的 IP 地址以 192 开始，而 Router 的 IP 地址以 193 开始，这表明这些节点属于不同的网络。在场景 2 中，在节点接口上指定的网络掩码导致不同的网络地址：192.168.10.0 和 192.168.10.2（注意 Server 的掩码值以 0 结束，而 Router 的掩码值以 192 结束，这将相应的地址映射到不同的网络）。在两种情形中，仿真将可能在没有异常终止的条件下执行，但 DES 日志将包含告警消息。在场景 1 中，错误日志消息将陈述，检测到不兼容的子网（即网络）地址，且在那些节点之间将没有流量发送，而在场景 2 中，错误日志消息将报告一个配置错误，陈述使用了不兼容的网络掩码。注意在两个场景中，DES 日志清晰地识别出错误配置发送的节点和接口，当调试仿真时，这是极其有帮助的。回顾一下，通过从 **Project Editor**（见 4.8 节）的下拉菜单中选择选

项 **DES→Open DES Log**（DES→打开 DES 日志），可访问 DES 日志。

表 9.1　不正确的接口配置例子

	节点 Server 中的接口 IF0		节点 Router 中的接口 IF10	
	IP 地址	掩码	IP 地址	掩码
场景 1	**192**. 168. 10. 1	255. 255. 255. 0	**193**. 168. 10. 2	255. 255. 255. 0
场景 2	192. 168. 10. 1	255. 255. 255. **0**	192. 168. 10. 2	255. 255. 255. **192**

9.3.4　IP 地址的自动指派

在许多情况下，手工指派 IP 地址是一件烦人的和容易出错的任务。将 IP 地址和掩码值设置为 Auto Assign 并让 OPNET 在仿真执行过程中处理 IP 寻址，经常是足够的。但是，在一些情形中，在仿真执行开始之前指派 IP 地址的做法，是人们希望这样做的。对于这个目的，OPENT 在 **Project Editor** 的 **Protocols** 下拉菜单中提供了一个选项集合。在本节，与陈述 IP 编址的下拉菜单选项的一个完整序列不同，如 **Protocols→IP→Addressing→Auto - Assign IPv4 Addresses**（协议→IP→寻址→自动指派 IPv4 地址），将仅指定菜单选项中的最后一个名字（如 **Auto - Assign IPv4 Addresses**）。

如图 9.6 所示，说明了通过 Protocols 菜单的所有可用的 IP 寻址选项，要在被仿真网络中的所有附接接口上自动指派 IPv4 地址，需要选择选项 **Auto - Assign IPv4 Addresses**。如果这项操作是成功的，那么 **Project Editor** 的底部平板将显示一条消息，说明指派了多少个地址。如果操作是不成功的，那么消息将说明 “No IP address assigned”（没有指派 IP 地址）。

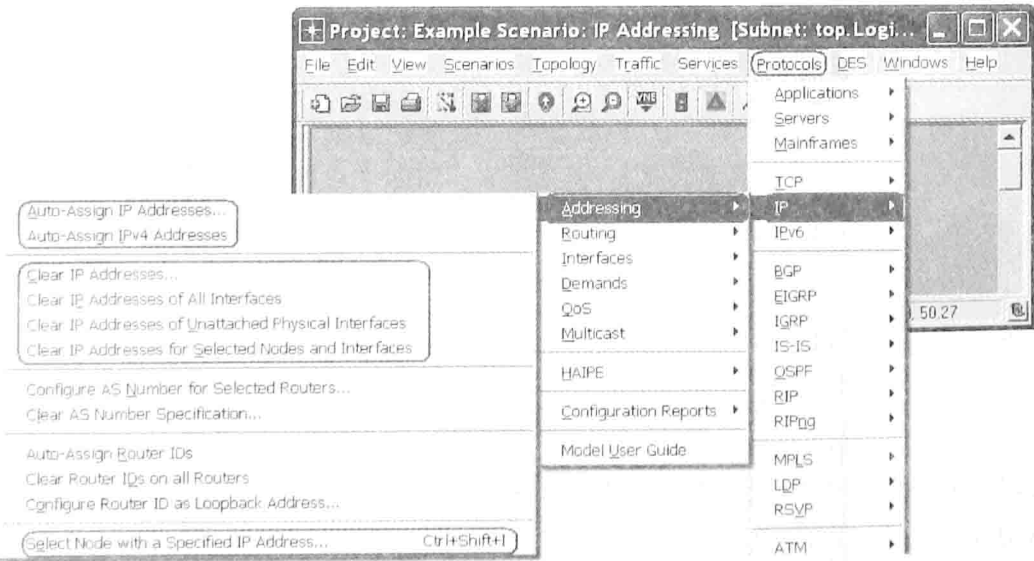


图 9.6　通过 Protocols 菜单可用的 IP 寻址选项

也可使用选项 **Auto - Assign IPv4 Addresses...** 自动指派 IP 地址。这个选项打开 **Auto - Assign IPv4 Addresses** 窗口，如图 9.7 所示，其中可为自动编址操作提供附加的

细节。具体而言，这个窗口将使用户可在如下配置选择中进行选择：

1）*Auto - assign IPv4 address*——自动指派 IPv4 地址。

2）*Set IPv4 addresses to* “*No IP Addresses*”（设置 IPv4 地址为“没有 IP 地址”）——禁止 IPv4 指派，从而网络中的各节点仅使用 IPv6 通信。选项 1）和 2）是相互排斥的，即在一个时刻仅可选择那些选项中的一个选项。

3）*Auto - assign global addresses*（*IPv6*）（自动指派全局地址（IPv6））——自动指派 IPv6 地址。

① *Exclude hosts*（排除主机）——IPv6 地址仅被指派到路由器，而端节点通过路由器通告发现它们的 IPv6 地址。这被称作 *stateless address auto - configuration*（无状态地址自动配置）。

② *Assign EUI - 64 addresses*（指派 EUI - 64 地址）——使用 IEEE 的扩展唯一标志符 - 64（EUI - 64）格式指派 IPv6 地址。EUI - 64 格式要求一个 IPv6 地址的 64 比特（即主机部分）基于相应的硬件地址。因为硬件地址通常是 48 比特的，所以 EUI - 64 格式以 FF - FF 或 FF - FE 填充其他的 16 比特，这取决于硬件地址是 MAC - 48 格式的（即网络硬件）或 EUI - 48 格式的（其他设备和软件）。如果没有选择这个检查框，那么 OPNET 指定非 - EUI - 64 IPv6 地址，即在不考虑硬件地址的条件下，产生整个 128 比特号码。

4）*All interfaces*（所有的接口）——将自动—地址配置应用到网络中的所有节点接口。

5）*Interfaces on selected nodes and links*（被选中节点和链路上的接口）——仅将配置应用到网络中的被选中对象。选项 4）和选项 5）是两两互斥的。为了使用选项 5），在尝试自动—指派地址之前，应该首先选择一些网络对象（节点和/或链路）。

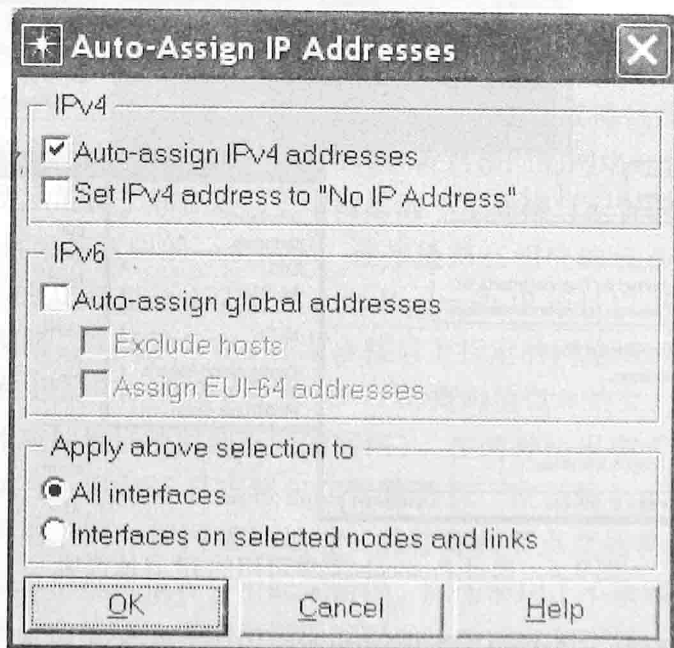


图 9.7 **Auto - Assign IPv4 Addresses** 窗口

9.3.5 清除IP地址指派

Protocols菜单页包含如下选项，用于自动地清除一个被仿真网络中的一个IP地址指派：

1）**Clear IP Addresses...** 打开 **Clear IP Addresses** 窗口，如图9.8所示，这使用户可配置清除地址指派操作的细节。通过这个窗口，可指定是否需要清除IPv4、IPv6或这两种地址类型，且是希望清除所有接口上的地址指派、仅是被选中节点和链路的各接口上的地址指派还是未附接物理接口上的地址指派。

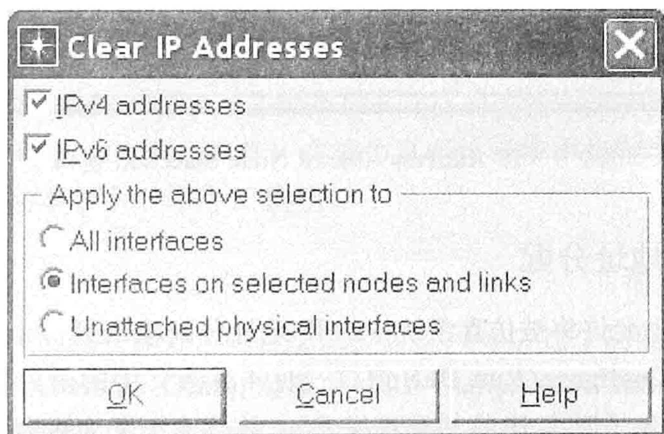


图9.8 **Clear IP Addresses** 窗口

2）**Clear IP Addresses of All Interfaces** 清除在网络中所有（附接的或非附接的）接口上的IPv4和IPv6地址。

3）**Clear IP Address of Unattached Physical Interfaces** 在网络中的所有非附接接口上清除IPv4和IPv6地址。

4）**Clear IP Addresses for Selected Nodes and Interfaces** 在被选中节点和链路的所有接口上清除IPv4和IPv6地址。

注意自动—指派操作仅将地址指派到附接的接口。因此，谨慎的做法是，仅在完成创建一个被仿真系统的网络拓扑过程之后才实施IP地址指派。如果网络拓扑已经改变，那么首先清除当前的IP地址，之后再实施自动—指派操作，就是一个不错的想法。

9.3.6 以一个指定的IP地址标志接口

OPNET也包含这样一个 **Protocols** 菜单选项，用于标志包含具有一个指定IP地址之接口的节点。可打开一个 **IP Address - based Node Selection**（基于IP地址的节点选择）窗口，如图9.9所示，方法是通过从 **Protocols** 菜单中选择 **Select Node with a Specified IP Address...** 或通过按下Ctrl + Shift + I键。这个工具似乎仅对IPv4地址有效。需要在相应文本框中指定一个IPv4地址，并单击 **Find** 按钮。需要单击 **Reload**（重新载入）按钮来刷新系统。如果被仿真系统没有包含带有指定IP地址的任何接口，那么将出现一

条警告消息。否则，带有一个节点表项列表的一个窗口将出现。每个节点表项由两个值组成：节点的名字和设置为指定 IP 地址值的接口名字。当正研究具有大量节点的一个仿真，并需要快速地定位具有一个特定 IP 地址时，这个功能是极其有用的。

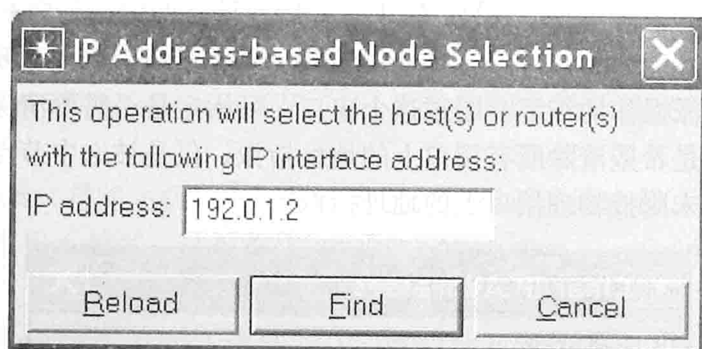


图 9.9 **IP Address - based Node Selection** 窗口

9.3.7 输出 IP 地址分配

最后，OPNET 也允许将被仿真系统中的 IP 地址分配输出到一个以逗号分隔的文本文件之中。可通过 **Configure/Run DES** 窗口（见 4.3 节）中的全局属性 **IP... IP Interface Addressing Mode**（IP... IP 接口寻址模式），将仿真配置为输出 IP 地址分配。这个属性接受如下四个值：

1）Auto Addressed（自动编址的）。在仿真过程中活跃接口上的所有 IP 地址都将被自动地指派，除非这些接口已经由手工进行了指派。OPNET 仅指派带有类别的地址（即类 A、B 或 C 中的地址）。指派到一个接口的 IP 地址类取决于在相应网络中主机的数量。接口的子网掩码是被指派地址类的默认掩码。

2）Auto Addressed/Export（自动编址的/输出）。除了自动地指派 IP 地址外，OPNET 也将它们输出到一个文件。文件被存储在默认模型目录，并依据如下惯例进行命名：<project - scenario - name - ip_ addresses. gdf>。

3）Manually Addressed（手工编址的）。仿真依赖于手工指派的 IP 地址。注意为了使仿真正确地运行，所有活跃的或可用的接口和作为最后方法的网关［即分别为端节点和核心节点中的属性 **Default Route**（默认路由）和 **Default Network**（s）（默认网络）］都必须有指派给它们的一个有效 IP 地址。但是，如果希望验证手工地址指派是正确的，那么应该将这个属性设置为 Auto Addressed 或 Auto Addressed/Export（即使地址是人工配置的也要这样做）。

4）Manually Addressed/Export（人工编址的/输出）。与 Manually Addressed 具有相同的含义，例外是在所有活跃接口上指派的 IP 地址被输出到一个文件。输出文件被命名和保存的方式，与这个属性被设置为 Auto Addressed/Export 时相同。

9.4　配置其他IP功能特征

在本节，描述在OPNET中被建模的其他IP功能特征的配置。在本节中描述的一些功能特征可通过节点属性进行配置，而其他的则通过IP工具对象进行配置。为指定场景范围的IP特征，OPNET包含工具对象的三个节点模型：

1）**IP Attribute Config**（IP属性配置）允许配置场景范围的IP功能细节，如压缩和ping探测。也可通过这个工具对象，配置仿真将路由表输入到外部文件。在本节后面描述由**IP Attribute Config**对象支持的IP功能特征。

2）**QoS Attribute Config**（QoS属性配置）允许为各种QoS机制指定配置。在第10章中通过**QoS Attribute Config**节点，讨论QoS支持的配置和部署。

3）**IP VPN Config**（IP VPN配置）在被仿真网络系统中指定节点之间的虚拟专用网（VPN）隧道。在本书中不讨论IP隧道。

9.4.1　IP压缩

通过压缩数据报的某些部分，IP压缩减少报文尺寸。这个功能特征在OPNET中得到支持。可通过**IP Attribute Config**对象中的**IP Compression Information**（IP压缩信息）属性，定义期望的压缩方案，之后将它们部署到选择的节点接口上。默认情况下，**IP Compression Information**属性包含一个预配置的压缩方案集合，包括TCP/IP首部压缩、每接口压缩、每虚拟电路压缩、图像压缩和Telnet应用压缩。但是，如果需要，可通过实施如下步骤，定义新的压缩方案：

1）将**IP Attribute Config**对象添加到项目工作空间中。如果**IP Attribute Config**对象已经被添加到当前场景中，则应该忽略这个步骤。

2）编辑**IP Attribute Config**对象的属性，方法是在其上右击，并选择**Edit Attributes**选项。

3）展开属性**IP Compression Information**。

4）将属性**Number of Rows**的值改变为一个大于6的数（因为存在6个预定义的压缩方案）。

5）展开新创建的IP压缩行之一，并通过指定如下属性值来定义新的压缩方案：

① **Name**为一个新压缩方案的唯一名字。这个值将在当前场景中识别这个IP压缩方案。

② **Compression Method**（压缩方法）指定压缩方法的类型。这个属性仅接受如下四个值之一：

- None（无）——无压缩。
- TCP/IP Header Compression（TCP/IP首部压缩）——仅有TCP/IP首部将被压缩。
- Per - Interface Compression（每个接口压缩）——整个报文将被压缩。

- Per - Virtual Circuit Compression（每个虚拟电路压缩）——仅有报文的净荷（即不是首部信息）将被压缩。

③ **Compression Ratio**（压缩率）和 **Compression Ratio PDF**（压缩率PDF）为概率分布函数（PDF）定义参数和PDF本身的名字。**Compression Ratio PDF** 计算被压缩报文和未被压缩报文之间的比率（典型地为0和1之间的一个值）。在Compression Ratio属性中指定的值被馈入到 **Compression Ratio PDF** 属性中定义的函数，后者接着计算被压缩报文和未被压缩报文之间的实际比率。OPNET仿真使用这个值计算被压缩报文的尺寸。

④ **Compression Delay（sec/bits）**［压缩延迟（秒/比特）］和 **Decompression Delay（sec/bits）**［解压缩延迟（秒/比特）］分别指定压缩和解压缩一条报文中单个比特所花费的时间。通过将以比特表示的报文尺寸乘以相应的（解）压缩延迟值，计算得到实际的压缩或解压缩延迟。

6）如有必要，可为许多压缩方案重复上述这个过程。

7）一旦定义了所有期望的压缩方案，则单击 **OK** 按钮保存所做的改变。

在定义压缩方案之后，就可通过设置属性 **Compression Information** 的值，将压缩方案部署在期望的接口上。可在端节点模型的 **IP... IP Host Parameters... Interface Information**（IP... IP主机参数 ... 接口信息）属性和核心节点模型的 **IP... IP Routing Parameters... Interface Information... <interface number>**（IP... IP主机参数 ... 接口信息 ... <接口号>）属性下找到这个属性。OPNET软件带有一个范例项目集合，展示说明了各种联网技术。这些项目位于example_ networks目录下。特别地，项目 **IP** 包含四个场景，它们展示了不同压缩方案对应用性能的影响。

9.4.2 路由协议的基本配置

OPNET包含如下路由协议的模型：路由信息协议（RIP）、开放最短路径优先（OSPF）、内部网关路由协议（IGRP）、增强的内部网关路由协议（EIGRP）、中间系统到中间系统（IS - IS）、边界网关协议（BGP）和几个移动自组织网络（MANET）协议。使用几个不同方法，可指定部署在网络中的路由协议，将在本节简短地讨论这些方法。典型情况下，每个网络接口有与之相关联的一个路由协议。通过设置 **Routing Protocol (s)**（路由协议）属性的值，可在一个核心节点的接口处指定路由协议。这个属性位于 **IP... IP Routing Parameters... Interface Information... <interface number>**（IP... IP路由参数 ... 接口信息 ... <接口号>）下。这个选项允许一次配置一个接口，当必须将路由协议部署到多个接口上时，这种做法是非常耗时的。

如图9.10所示的 **Routing Protocol Configuration**（路由协议配置）窗口，提供了一次在网络中的多个接口上部署路由协议的一种方法。可通过 **Protocols→IP→Routing→Configure Routing Protocols**（协议→IP→路由→配置路由协议）选项打开这个窗口。**Routing Protocol Configuration** 窗口使您可指定信息，如要部署的路由协议的名字［即RIP、IGRP、OSPF、IS - IS、EIGRP或None（无）］，路由协议是部署在所有接口上还是

仅部署在那些被选中的链路上，以及最后，路由域是否要进行可视化处理。通过单击 **OK** 按钮，可保存路由协议配置，并在指定的接口上重写 **Routing Protocol（s）**属性的值。

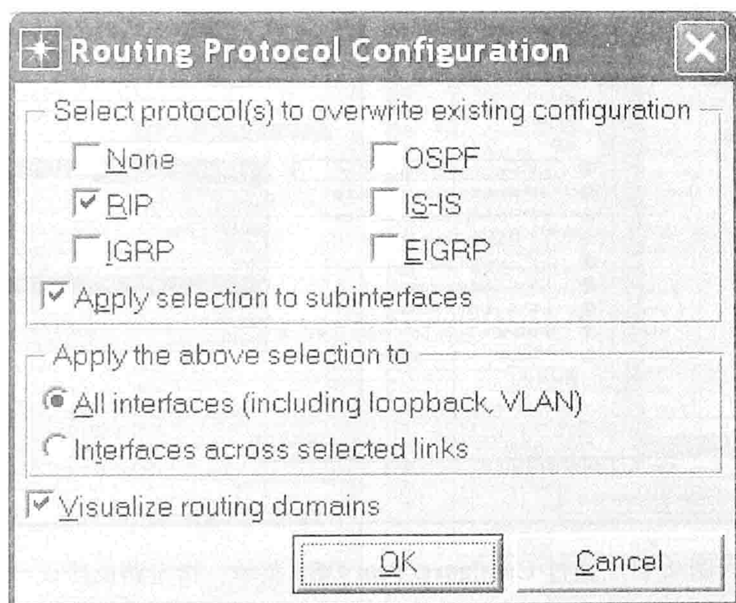


图 9.10 **Routing Protocol Configuration** 窗口

最后，如图 9.11 所示，通过设置全局属性 **IP... IP Dynamic Routing Protocols**（IP... IP 动态路由协议），也可指定要部署在被仿真系统中的路由协议的名字。当这个属性被设置为 Default 时，仿真使用在个体接口上本地配置的路由协议。在所有其他情形中，所指定的路由协议要优先于每个接口上本地指定的协议而被采用。唯一的例外是当一个接口运行 IPv6 时的情况。在这样一种情形中，将使用一个本地定义的非—MANET 路由协议，而不是通过 **IP... IP Dynamic Routing Protocol** 属性指定的全局值。也可通过在各种核心节点模型处改变属性 IP Routing Protocols 的值，指定路由协议的配置细节。第 11 章比较详细地讨论路由协议配置。

9.4.3 配置 IP 接口的不同类型

OPNET 节点模型支持如下接口类型：物理（physical）、环回（loopback）、隧道（tunnel）、聚合（aggregate）和子接口（subinterface）。典型情况下，分别通过核心节点和端节点模型中的属性 **IP... IP Routing Parameters... Interface Information...** 和 **IP... IP Host Parameters... Interface Information** 配置物理接口，在 9.2.1 节和 9.2.2 节描述。

环回接口仅存在于核心节点模型。默认情况下，每个核心节点模型定义了一个环回接口。通过检查 **IP... IP Routing Parameters... Loopback Interfaces** 属性，可查看环回接口配置。通过展开属性 **Loopback Interfaces**，将其子属性 **Number of Rows** 的值设置为期望的环回接口的总数，之后配置每个个体新添加的环回接口，这样就可创建附加的

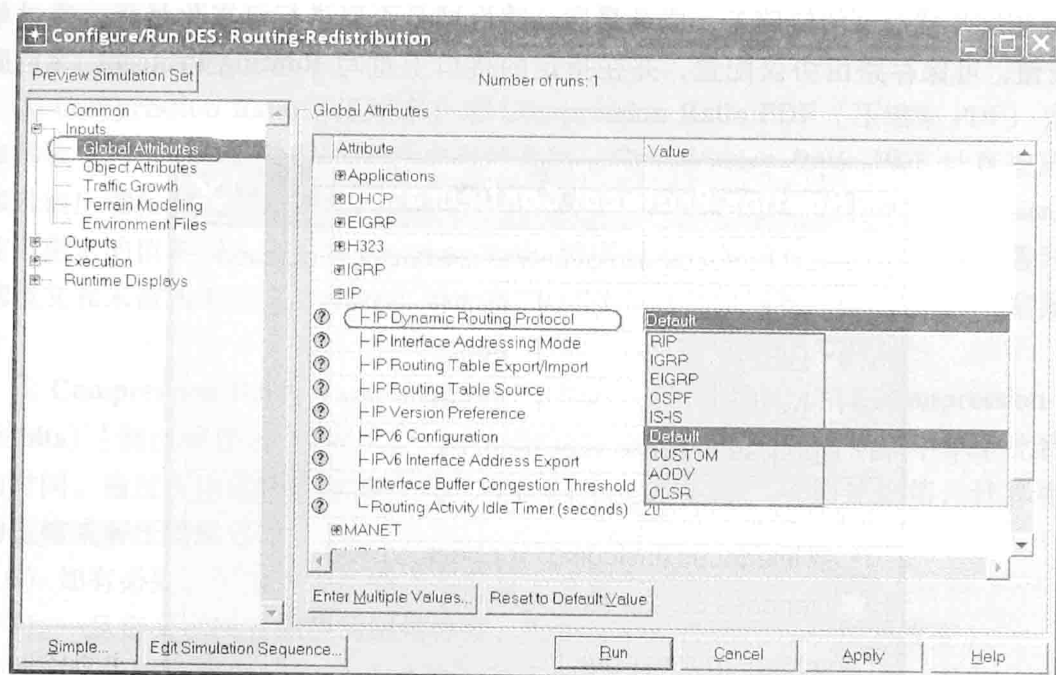


图9.11 通过**Configure/Run DES**菜单，指定路由协议

环回接口。也可以通过**Protocols→IP→Interfaces→Create Loopback Interface...**（协议→IP→创建环回接口...）选项创建环回接口。

类似地，通过设置**IP...IP Routing Parameters...Tunnel Interfaces**（IP...IP路由参数...隧道接口）属性的值，可创建隧道接口。也仅在核心节点模型中支持隧道接口。但是，核心节点模型不包含任何预设置的默认隧道接口。通过展开属性**Tunnel Interfaces**，将其子属性**Number of Rows**的值设置为期望的隧道接口总数，之后配置每个个体新的附加隧道接口，就可添加期望数量的隧道接口。

OPNET也支持接口聚合，这是将多个物理接口映射到单个逻辑接口的一种机制。指定接口聚合的过程由两个不同的步骤组成：定义聚合接口和将物理接口映射到定义好的聚合接口。可通过属性**IP...IP Routing Parameters...Aggregate Interfaces**（IP...IP路由参数...聚合接口）定义聚合接口。同样，首先需要将子属性**Number of Rows**的值设置为要配置的聚合接口总数，之后配置每个个体新创建的聚合接口。每个聚合接口包含一个配置属性集合，这与用来配置物理接口的那些属性相同。属性**Name**唯一地标志每个聚合接口，并被用来将物理接口映射到聚合接口，这是过程中的下一步。为了指定属于一个特定聚合接口的一个物理接口，需要将属性**IP...IP Routing Parameters...Interface Information...<interface number>...Aggregation Parameters...Aggregate Interface**（IP...IP路由参数...接口信息...<接口号>...聚合参数...聚合接口）的值设置为聚合接口的名字，如图9.12所示。

最后，核心节点模型也支持设置子接口，这是允许单个物理接口分隔为几个逻辑子接口的一种机制。OPNET为如下协议实现子接口功能：ATM、FR、串行和VLAN。通过

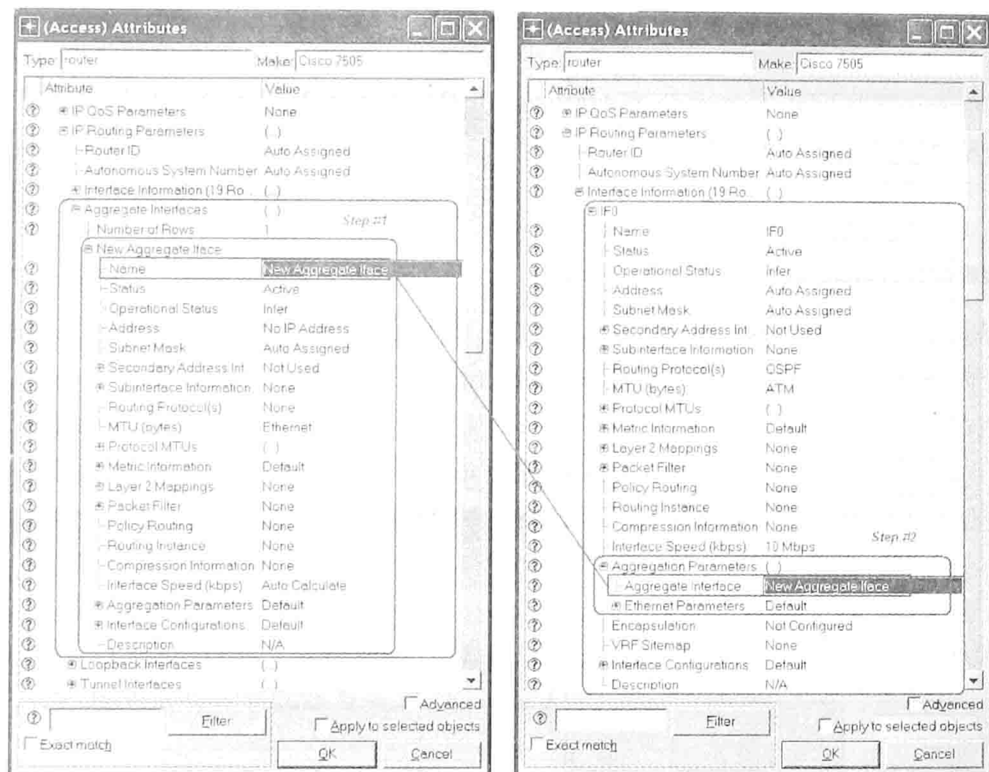


图 9.12　配置聚合接口的步骤

设置属性 **IP... IP Routing Parameters... Interface Information... < interface number > ... Subinterface Information**（IP... IP 路由参数... 接口信息... <接口号>... 子接口信息）的值，可配置子接口。同样，将需要设置属性 **Number of Rows** 的值，之后配置每个新创建的子接口。典型情况下，依据如下惯例命名子接口：其物理接口的名字后跟一个句号和子接口号。例如，如果为接口 IF12 定义三个子接口，那么这些子接口应该命名为 IF12.1、IF12.2 和 IF12.3。另外，OPNET 文档提醒您不能将一个子接口的 IP 地址设置为 Auto Assigned，相反必须显式地输入 IP 地址值。

一旦指定基本的子接口信息，则在附接到这个子接口的 ATM/FR 或 VLAN/ELAN 标志符情形中，需要指定一个永久虚电路（PVC）名。通过如图 9.13 所示的子属性 Layer 2 Mapping（层 2 映射），可指定 PVC 名或标识符号。注意，ATM 和 FR 的 PVC 是分别通过工具对象 **ATM_ SPVx_ Config** 和 **FR PVC Config** 定义的。另外，可通过 **Protocols→IP→Interface→Create sub-interfaces for selected ATM/FR PVXs...**（协议→IP→接口→为选中的 ATM/FR PVXs 创建子接口...）指定子接口。在 example_ networks 目录中的范例项目 IP 包含称为 Subinterfaces 的一个场景，它提供了子接口配置的一个例子。

9.4.4　配置 IP 负载均衡

存在到一个目的地的多条路由情况下，可将一台路由器配置为实现负载均衡，这是为优化网络利用率将流量分布在多条路径上的一种技术。可通过属性 **IP... IP Routing**

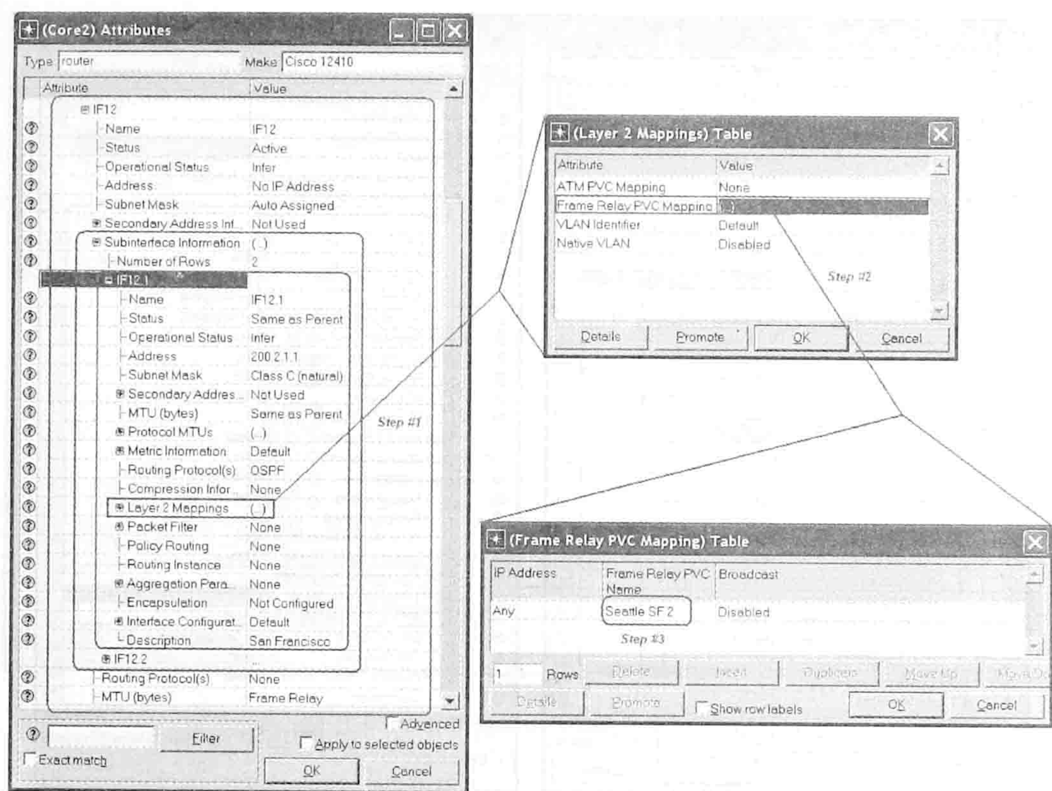


图 9.13　配置子接口的步骤

Parameters... Load Balancing Options（IP... IP 路由参数... 负载均衡选项）配置负载均衡。这个属性接受如下两个值，这些值表示由 OPNET 支持的负载均衡类型：

1）Destination - Based（基于目的地的），依据源—目的地 IP 地址对分配流量。在这种方法中，由源—目的地 IP 地址对识别的属于同一流量断续流的所有报文，在它们到目的地的路上被保障遵循同一路由。这是负载均衡技术的默认设置。属于不同流的报文可能在不同路径上被路由，即使这些流可能到相同目的地也是如此。

2）Packet - Based（基于报文的），在每报文基础上分配流量——后续报文在到目的地的其他路径上转发，其中路径是以一种轮转方式被选择的。这种方法在多条路径上提供精细粒度的负载分配，但它可能导致属于同一条流的各报文的报文重新排序，对于某些应用而言这也许是不可接受的。

9.5　互联网控制消息协议

互联网控制消息协议（ICMP）经常与 IP 一起讨论，原因是它解决如错误报告和查询管理等 IP 的缺陷。在如今网络中存在 ICMP 的两个版本：ICMPv4 和 ICMPv6。但是，ICMPv6 得到 OPNET 的系统在环路内（SITL）产品的支持，该产品是 OPNET Moderler 版本 15.0 或后续版本的一个附加模块。由于篇幅限制，在本书中不讨论 SITL 和 ICMPv6，这就是在本章中将 ICMPv4 简单地称为 ICMP 的原因。

由 OPNET 支持的唯一 ICMP 类型是 ICMP 回声请求和回声应答消息，这些消息是通过 IP ping 需求对象建模的。但是，多数路由器节点模型包含允许激活或禁止某些 ICMP 消息的属性。这些属性是在 **Interface Configurations** 复合属性之下定义的，并典型地用于验证路由器配置的正确性，这超出了本书的范围。本节的后面部分讨论部署 IP ping 需求的步骤，并详细研究各种 ping 统计量。

9.5.1　指定 Ping 模式

每个 IP Ping 需求对象都包含一个称为 **Ping Pattern**（Ping 模式）的属性，该属性指定 ping 流量的各种特征。可通过 **IP Attribute Config**（IP 属性配置）工具对象的 **IP Ping Parameters**（IP Ping 参数）属性定义一个 ping 模式。OPNET 已经提供了一个预配置的 ping 模式集合，但可通过执行如下步骤定义自己的模式：

1）将一个 **IP Attribute Config** 对象添加到项目工作空间中，如果该对象还不存在于当前仿真场景内的话。

2）打开这个对象的 **Attributes** 窗口。

3）展开属性 **IP Ping Parameters**，如图 9.14 所示。

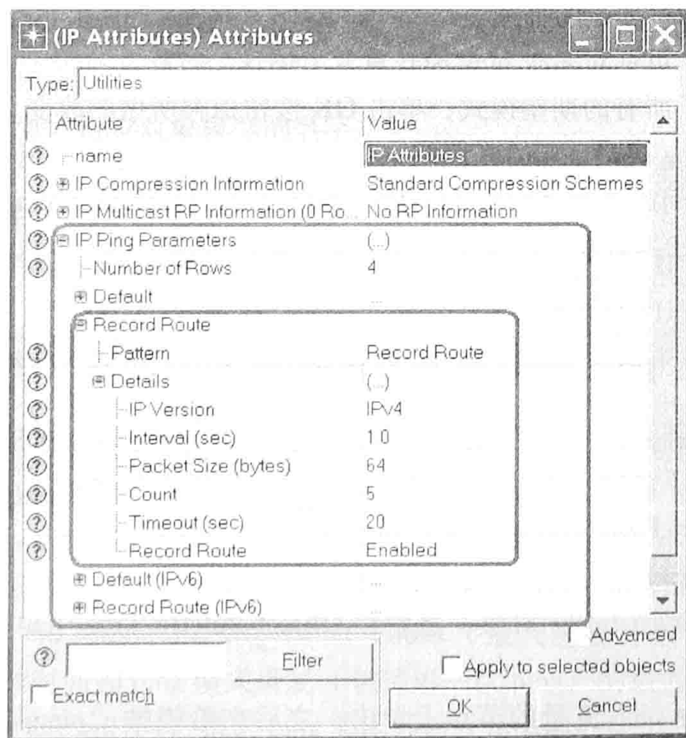


图 9.14　**IP Ping Parameters** 属性

4）将属性 **Number of Rows** 的值改变为大于 4 的一个数（默认条件下，存在 4 个预定义的 ping 模式）。

5）展开一个新创建的行，并通过指定如下属性的值来定义 ping 新模式：

① **Pattern**（模式）——ping 模式的场景范围内的唯一名字。

② **Details**（细节）——一个复合属性，为 ping 模式指定流量细节。它由如下子属性组成：

- **IP Version**（IP 版本）指定由报文（携带 ICMP 消息）使用的 IP 版本。这个属性仅支持两个可能的值：IPv4 和 IPv6。
- **Interval**（**sec**）［间隔（秒）］指定连续 ping 消息之间的时长（s）。
- **Packet Size**（**bytes**）［报文尺寸（字节）］指定 ping 报文的尺寸。这个值不包括 8 字节的 ICMP 首部，在首部是每条 ping 消息被封装在 IP 报文内时添加到消息上的。
- **Count**（计数）指定由这个模式产生的 ICMP 回声—请求报文的数量。
- **Timeout**（**sec**）［超时（秒）］指定以 s 为单位的节点将等待的时长，如果没有来自被 ping 节点的响应，它认为 ICMP 消息丢失之前将等待的时长。
- **Record Route**（记录路由）激活或禁止记录 ping 消息所取路由的选项。如果 ping 需求被配置带有这样一个 ping 模式，该模式激活了 **Record Route** 属性，那么将记录 ICMP 回声请求和回声应答消息所取的路径。通过详细研究 **DES Run Tables**（DES 运行表）［通过 **View Results**（查看结果）选项］，可在仿真完成时查看所记录的路径。所记录的路由是以一个表的形式报告的，该表有如下各列：*IP address*（IP 地址）、*Hop Delay*（跳延迟）、*Node Name*（节点名）、*MPLS Label*（MPLS 标签）和 *MPLS EXP*。

6）如有必要，可针对多次 ping 模式重复上述这个过程。

7）一旦添加了所有的期望模式，单击 **OK** 按钮保存所做的改变。

一旦定义了 ping 模式，则可在网络中部署 IP ping 需求。通过拖放 *ip_ ping_ traffic* 对象到项目工作空间或使用 **Protocols→IP→Demands→Configure Ping Traffic on Selected Nodes**（协议→IP→需求→在被选中的节点上配置 ping 流量），可指定 IP ping 需求。在 9.5.2 节和 9.5.3 节中详细研究这些方法中的每种方法。

9.5.2 利用 ip_ ping_ traffic 对象部署 IP Ping 需求

采用这项技术部署一个 IP Ping 需求的过程，第一步是将必要的需求对象添加到项目工作空间中。为了添加一个 IP Ping 需求，需要实施如下步骤：

1）打开 **Object Palette Tree**（对象调色板树），并找到 *ip_ ping_ traffic* 对象（位于 *Demand Models*、*demands* 和 *internet_ toolbox* 文件夹内）。

2）选择 *ip_ ping_ traffic* 对象，之后在 **Object Palette Tree** 中单击 *ip_ ping_ traffic* 对象的图标。

3）在希望产生 ping 流量的节点上单击，之后在希望被“pinged”的节点上单击。结果将是连接这两个节点的一条点画线，代表 IP 流量需求对象。

4）在任意成对节点之间可创建许多这样的对象。当完成时，右击项目工作空间，并从弹出菜单中选择 **Abort Demand Definition**（中止需求定义）。

如果希望去除一个需求对象，那么选择该对象，并按 Delete（删除）键。

一旦添加了所有的期望 IP ping 需求，则可能希望修改它们的配置。下面是对所有 IP ping 需求属性的一个简短描述，如图 9.15 所示：

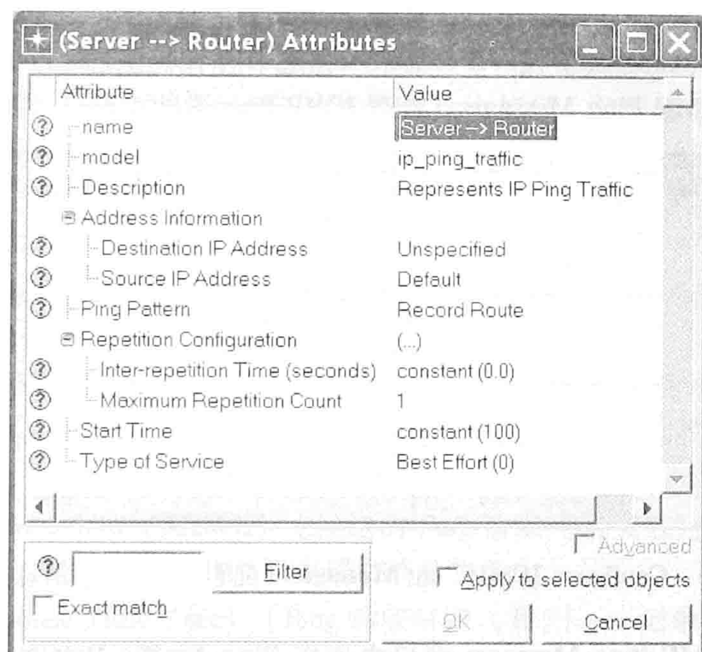


图 9. 15　一个 IP ping 需求对象的属性

1）**name** 指定 IP ping 需求对象的名字。

2）**model**（模型）指定对象模型的名字。

3）**Description**（描述）提供有关这个需求对象之目的的一个简短提示。这个属性对仿真执行没有影响。

4）**Address Information**（地址信息）通过子属性 **Destination IP Address**（目的地 IP 地址）和 **Source IP Address**（源 IP 地址），指定需求之源和目的地节点的 IP 地址。

5）**Ping Pattern**（Ping 模式）指定在 **IP Attribute Config** 对象中定义的 ping 模式的名字。

6）**Repetition Configuration**（重复配置）指定这个需求的重复模式。这个复合属性有如下预设值：Once at Start Time（在开始时刻重复一次）、Every 10 minutes（每隔 10 分钟）、Every 30 minutes（每隔 30 分钟）和 Every 1 hour（每隔 1 小时）。如果期望的话，可定义自己的重复模式，方法是设置子属性 **Inter - repetition Time**（**seconds**）[重复间隔时间（秒）] 和 **Maximum Repetition Count**（最大重复计数），分别指定连续 ping 模式重复之间的时间和 ping 模式重复（包括第一次 ping）的最大数量。

7）**Start Time**（开始时间）指定 ping 需求开始发送 ICMP 消息的时间。

8）**Type of Service**（服务类型）指定将携带 ICMP 消息的 IP 报文的 ToS 值。

一旦配置了所有期望的需求对象，那么就完成了部署 IP ping 需求的过程。在此之后，仅需要选择要在仿真过程中收集的 ping 统计量，将在 9. 5. 4 节讨论。

9. 5. 3　使用协议菜单部署 IP Ping 需求

在通过 **Protocols** 菜单部署 IP ping 需求之前，需要至少选择将参与 IP ping 流量交换

的两个节点。一旦选择期望的节点，选择 **Protocols→IP→Demands→Configure Ping Traffic on Selected Nodes**（协议→IP→需求→在选中的节点上配置 Ping 流量）选项，这打开 **Configure ICMP Ping Messages**（配置 ICMP Ping 消息）窗口，如图 9.16 所示。

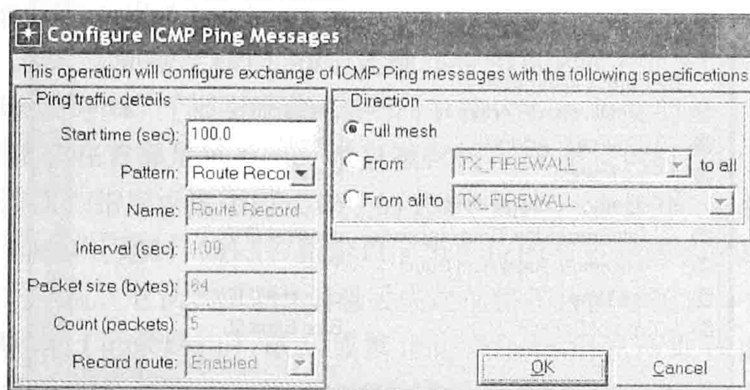


图 9.16　**Configure ICMP Ping Messages**（配置 ICMP Ping 消息）窗口

Configure ICMP Ping Messages 窗口由称作 **Ping Traffic Details**（Ping 流量细节）和 **Direction**（方向）的两个平板组成。默认情况下，**Ping Traffic Details** 平板仅激活 **Start time**（**sec**）和 **Pattern** 属性。**Pattern** 属性的下拉文本框包含通过 **IP Attribute Config** 对象定义的所有 ping 模式的列表和一个选项 New...，当选择后者时，将创建一个新的 ping 模式，并将激活其他的属性。这些属性与在 **IP Attribute Config** 对象中定义 ping 模式的那些属性相同，因此省略了其描述。

注意，**Configure ICMP Ping Messages**（配置 ICMP Ping 消息）窗口不允许配置 ping 需求重复性、ToS 和地址信息。通过 **Protocols** 菜单部署的所有 ping 需求都将在 **Start time**（**sec**）中指定的时刻执行一次，且携带 ICMP 消息的 IP 报文将以默认 ToS 值 0 进行设置。

Configure ICMP Ping Messages 窗口的 **Direction** 平板被用来配置一个 ping 需求的源和目的地地址信息。如图 9.16 所示，**Direction** 平板有如下三个选项，它们指定 ping 需求将如何部署在被选中的节点间：

1）**Full mesh**（全互联）。每个被选中的节点都将 ping 需求流量发送到所有其他被选中的节点，并将从所有其他被选中的节点接收 ping 需求流量。

2）**From < pull - down textbox > to all**（从 < 下拉文本框中的节点 > 到所有节点）。这个选项允许从单个节点产生到所有其他被选中节点的 ping 流量。下拉文本框包含被选中的节点名列表。在下拉文本框中被选中的节点将作为 ping 流量的一个源。

3）**From all to < pull - down textbox >**（从所有节点到 < 下拉文本框中的节点 >）。这个选项允许产生从所有其他被选中节点到单个节点的 ping 流量。下拉文本框包含被选中的节点名列表。在下拉文本框中被选中的节点将作为 ping 流量的一个目的地。

一旦完成 ping 需求的配置，则需要单击 **OK** 按钮保存所做的改变并部署指定的需求。如果没有使用在 **IP Attribute Config** 中定义的 ping 模式之一，相反却创建一个新

的，那么新的ping模式将作为一个新行被添加在**IP Attribute Config**对象中**IP Ping Parameters attribute**下。注意，即使**Configure ICMP Ping Messages**窗口不包括配置重复模式和ToS值的属性，仍然可采用如下方法指定这个信息，首先通过**Protocols**菜单创建ping需求，之后在**IP Attribute Config**对象中改变其对应的属性值。

9.5.4 Ping统计量

在执行仿真之前，也许希望选择要在仿真运行过程中收集的Ping统计量。在节点统计量的IP类中可找到Ping特定的统计量：

1）**Ping Replies Received**（**packets**）[接收到的Ping应答（报文数）]——记录作为回声请求消息（发送到一个特定目的地）的响应而在一个节点接收到的ICMP回声应答消息数量。

2）**Ping Request Sent**（**packets**）[发送的Ping请求（报文数）]——记录由一个节点发送到一个特定目的地的ICMP回声请求消息数量。

3）**Ping Response Time**（**sec**）[Ping响应时间（秒）]——记录自一条ICMP回声请求被发送直到接收到相应的ICMP回声应答所经过的时间。

最后，回顾一下，对于将其ping模式的**Record Route**（记录路由）属性设置为Enabled的任何IP ping需求，将记录由其ICMP回声请求和回声应答消息所经过的路径。通过打开**Results Browser**中的**DES Run Table**（DES运行表）选项卡，可查看所记录的路由。同样，OPNET在example_ networks目录中项目**IP**的icmp_ route_ pint场景中提供了这种功能特征的一个绝佳展示，IT Guru和Modeler软件的所有发行版中都带有标准example_ networks目录。

9.6 常用IP统计量、表和报告

9.6.1 IP统计量

OPNET提供了详细研究IP性能的一个统计量集合。具体而言，在如下节点统计量类中存在IP统计量：**IGMP Host**（ICMP主机）、**IGMP Router**（IGMP路由器）、IP、**IP Interface**（IP接口）、**IP Processor**（IP处理器）、**IP Tunnel**（IP隧道）、**IP VPN tunnel**（IP VPN隧道）、**IPv6**、**PIM－SM**、**Route Table**（路由表）、**Router Convergence**（路由器收敛）以及路由协议的几个其他类。类似地，IP全局统计量类包括**IP**、**IPv6**、**PIM－SM**和路由协议类。在本节，仅提供通用IP功能之统计量的简要描述。这里在讨论中略去与特定IP模块有关的统计量或在本书的相应部分描述，这里统计量诸如IP组播（即**IGMP Host**、**IGMP Router**、来自IP和IPv6类的一些统计量和**PIM－SM**）、QoS（即**IP Interface**）、IPv6（即**IPv6**）、ICMP（即来自**IP**类的某些统计量）、IP处理器（即**IP Processor**）、VPN（即**IP VPN tunnel**）和各种路由协议。表9.2提供通用IP功能的统计量概述。

表9.2 通用IP功能特征的统计量

类	名字	描述
Global Statistics IP（全局统计量IP）	*Background Traffic Delay (sec)*（背景流量延迟）	记录穿越网络的背景流量所经历的端到端延迟
	Network Convergence Activity（网络收敛活动）	记录网络中的收敛活动（即所有路由的转发表中的变化）。当存在收敛活动，表示为1的时段；当没有收敛活动，表示为0的时段
	Network Convergence Duration (sec)[网络收敛时长（秒）]	记录网络中的所有路由器的IP转发表收敛（即转发表不再经历变化）所花费的时间
	Number of Hops（跳数）	记录IP报文到达其相应目的地所穿越的平均跳数
	Traffic Dropped (packets/sec)[丢弃的流量（报文数/秒）]	记录由网络中所有节点在其所有接口上丢弃的IP报文总数
Node Statistics IP（节点统计量IP）	*Background Traffic Delay (sec)* ←[背景流量延迟（秒）←] *Background Traffic Delay (sec)* →[背景流量延迟（秒）→]	记录背景流量断续流所经历的端到端延迟（即一个单位的背景流量数据从流源传输到流目的地所花费的时间）。这些统计量是分别针对到达这个节点（即这个节点是流的目的地）的所有背景流或从这个节点离开（这个节点是流的源）记录得到的
	Background Traffic Flow Delay (sec)[背景流量断续流延迟（秒）]	记录由背景流量断续流经历的端到端延迟。这个统计量是在每个流基础上针对源自这个节点的流记录得到的
	Broadcast Traffic Received (packets/sec)[接收到的背景流量（报文数/秒）] *Broadcast Traffic Sent (packets/sec)*[发送的背景流量（报文数/秒）]	记录由这个节点在其所有接口上接收或发送的广播报文数。这些统计量是以报文数/s为单位记录的
	End-to-end Delay (sec)[端到端延迟（秒）]	记录一条报文的端到端延迟。端到端延迟的计算：计算报文到达目的地的时间和报文被创建的时间之差。这个统计量是仅针对单播流量记录的。为每个源—目的地对计算一个单独的统计向量

（续）

类	名字	描述
Node Statistics IP（节点统计量 IP）	*End - to - end Delay Variation*（*sec*）[端到端延迟方差（秒）]	记录当报文到达目的地时所经历的端到端延迟方差或抖动。这个统计量仅针对单播流量进行记录。为每个源—目的地对计算一个单独的统计向量
	Number of Hops←（跳数←） *Number of Hops→*（跳数→）	记录报文穿越的跳数，这些报文是到达该节点的（即源自多个不同源），或从这个节点离开的（即传输到多个目的地）。在后一种情形中，当目的地节点从这个节点接收到报文时，它们为这个节点记录统计量
	Processing Delay（*sec*）[处理延迟（秒）]	记录一条报文经历的 IP 层处理延迟。从报文到达 IP 层的时间直到该报文离开 IP 层的时间计算延迟值
	Traffic Dropped（*packets/sec*）[丢弃的流量（报文数/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	这些统计量记录在这个节点所有接口上丢弃的、接收到的和发送的 IP 流量总量（即单播、组播和广播）
Node Statistics IP Tunnel（节点统计量 IP 隧道）	*Delay Variation*（*sec*）[延迟方差（秒）]	记录报文穿越隧道所经历的延迟方差或抖动
	ETE Delay（*sec*）[*ETE* 延迟（秒）]	记录报文穿越隧道所经历的端到端延迟。记录的值包括封装延迟、报文穿越隧道所花费的时间和解封装延迟
	Traffic Dropped（*bits/sec*）[丢弃的流量（比特/秒）] *Traffic Dropped*（*packets/sec*）[丢弃的流量（报文数/秒）]	以 bit/s 和报文数/s 为单位记录在 IP 隧道接口上的流量丢弃率。得到的统计量的名字是在其后附加一个隧道的名字
	Traffic Received（*bits/sec*）[接收到的流量（比特/秒）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）]	以 bit/s 和报文数/s 为单位记录在 IP 隧道接口上的流量到达率。得到的统计量的名字是在其后附加一个隧道的名字
	Traffic Sent（*bits/sec*）[发送的流量（比特/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	以 bit/s 和报文数/s 为单位记录在 IP 隧道接口上的流量发送率。得到的统计量的名字是在其后附加一个隧道的名字

（续）

类	名字	描述
Node Statistics IP VPN Tunnel（节点统计量IP VPN隧道）	*Packets Received*（*packets*）[接收到的报文（报文数）] *Packets Received*（*packets/sec*）[接收到的报文（报文数/秒）]	分别以报文数和报文数/s为单位记录从这个VPN隧道接收到的平均报文数和总数
	Packets Sent（*packets*）[发送的报文（报文数）] *Packets Sent*（*packets/sec*）[发送的报文（报文数/秒）]	分别以报文数和报文数/s为单位记录在这个VPN隧道上发送的平均报文数和总数
	Tunnel Delay（*sec*）[隧道延迟（秒）]	记录报文穿越这条VPN隧道所经历的平均端到端延迟
Node Statistics Route Table（节点统计量路由表）	*Number of Next Hop Updates*（下一跳更新次数）	针对一条特定路由，在IP转发表中下一跳字段值被改变的次数
	Number of Route Additions（路由添加的次数）	记录一条路由从IP转发表中被添加的次数
	Number of Route Deletions（路由被删除的次数）	记录一条路由从IP转发表中被删除的次数
	Size（*number of entries*）[尺寸（表项数）]	记录IP转发表的尺寸。默认情况下，这个统计量是以桶（bucket）模式使用求和函数收集的。因此，针对每个桶周期的时长，记录IP转发表的最小、最大和平均尺寸
	Time Between Updates（*sec*）[更新之间的时间（秒）]	记录IP转发表的两次连续更新之间的平均时间长度
	Total Number of Updates（更新总数）	记录在这个节点IP转发表被更新的次数
Node Statistics Router Convergence（节点统计量路由器收敛）	*Convergence Activity*（收敛活动）	记录在路由器处的收敛活动（即在转发表中发生的改变）。存在收敛活动的时间段表示为1，没有收敛活动的时间段表示为0
	Convergence Duration（*sec*）[收敛时长（秒）]	记录在这个路由器处IP转发表收敛到一个稳定状态所花费的时间（即直到转发表不再有更多的改变）。每个被记录的值对应于IP转发表到达一个收敛状态所花费的时间

9.6.2　可视化和配置报告

OPNET 提供了几个其他有用的选项，可帮助更好地理解一个被仿真 IP 网络的配置和性能。具体而言，OPNET 通过 **Project Editor** 中 **View→Visualize Protocol Configuration**（查看→可视化协议配置）、**View→Visualize Network Configuration**（查看→可视化网络配置）和 View→Visualize Link Usage（查看→可视化链路使用情况）等下拉菜单，支持 IP 层的各种功能特征和其他协议配置的可视化。这些选项是非常容易使用的，可提供某些网络配置功能特征的一个快照视图，在调试过程中是非常有用的。

也可产生各种网络配置和性能报告。具体而言，可通过 **Protocols→IP→Configuration Reports→Select/Generate...**（协议→IP→配置报告→选择/产生...）选项产生配置报告，这个选项打开一个 **Generate Selected Configuration reports**（产生选中的配置报告）窗口，如图 9.17 所示。通过在那个窗口内选择检查框，可选择要产生哪些配置报告。另外的方法是，可选择 **Protocols→IP→Configuration Reports→Generate All**（协议→IP→配置报告→产生所有报告）选项，这将产生所有可用的报告。通过 **Results Browser** 窗口中的 **Configuration Reports Tables**（配置报告表）选项卡，可查看所产生的报告。

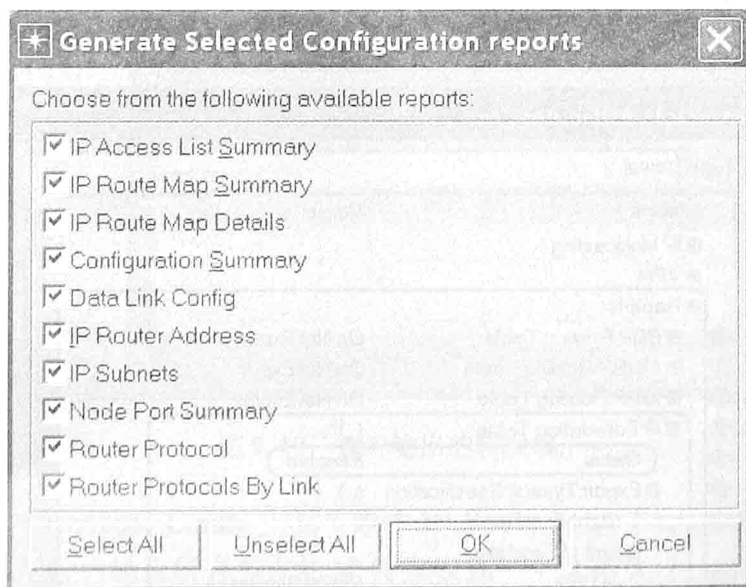


图 9.17　**Generate Selected Configuration reports**（产生被选中的配置报告）窗口

9.6.3　查看转发表和路由表

查看在一次仿真过程中在各节点处产生的 IP 转发表和路由表，经常是重要的事情。在一个节点的接口处部署的每个路由协议都产生一个路由表，它存储有关这个节点处路由协议已知的路由的信息。IP 将该节点处所有的这些路由表拉到那个节点的一个通用 IP 转发表（有时也称为 IP 通用路由表）。IP 使用 IP 转发表，为到达那个节点的每条报

文做出转发决策。OPNET 文档在对属于“转发表”的使用方面是不一致的，原因是这个术语在许多地方也包括路由表，而在其他地方，术语“路由表”又被用来指代 IP 转发表。在本书中，使用术语 IP“转发表”仅指代 IP 的转发表，将由路由协议使用的表称为路由表。

可通过 **Reports** 属性配置节点输出各种转发/路由表。这个属性存在于多数核心节点和边缘节点模型中，并配置节点输出各种路由表，包括 IP 转发表。这里详细研究如何输出在一个节点中产生的 IP 转发表。输出由特定路由协议产生的路由表的步骤是非常类似的，为避免冗余，略去它们的描述。

输出 IP 转发表的最简单方式，是在各节点处改变 **Reports... IP Forwarding Table**（报告... IP 转发表）属性的值，其中在这些节点处希望详细研究其 IP 转发表。这个属性具有如下预定义值：

1）Do Not Export（不输出）指明将不输出 IP 转发表。

2）Export at End of Simulation（在仿真结束时输出）指明在仿真完成时将输出在这个节点处产生的 IP 转发表。

3）Edit... 使您可指定在仿真过程中要输出 IP 转发表的时间。图 9.18 给出这个属性配置的一个例子，使在仿真结束和在时刻 10s 时输出 IP 转发表。注意，为了输出 IP 转发表，必须将 **Reports... IP Forwarding Table... Status**（报告... IP 转发表... 状态）属性的值设置为 Enabled。

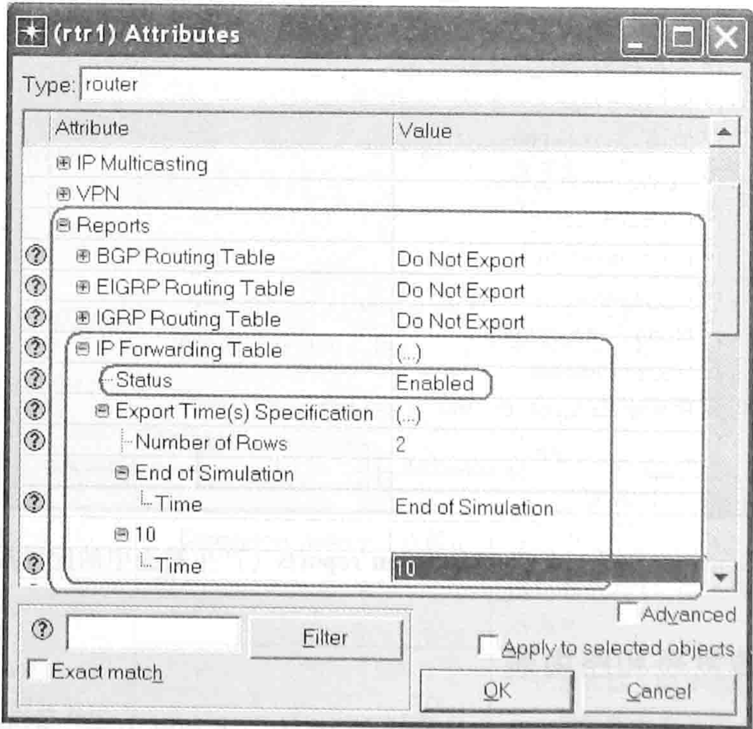


图 9.18 **IP Forwarding Table**（IP 转发表）属性

也可通过选择 **Protocols→IP→Routing→Export Routing Tables...**（协议→IP→路

由→输出路由表...）选项，输出 IP 转发表。这个选项打开一个窗口，其中可指定希望输出网络中所有节点还是仅有被选中节点（如果已经提前选中一些节点）的 IP 转发表。当单击 OK 按钮应用选择时，则在被选中节点上 **Reports... IP Forwarding Table** 属性的值将被改变为 Export at End of Simulation。另外，将出现一个窗口，通知已经设置了多少个 **IP Forwarding Table** 属性。最后，通过 **Configure/Run DES** 窗口，在所有节点上可将仿真配置为输出 IP 转发表。将全局属性 **IP... IP Routing Table Export/Import** 的值设置为 Export，将仿真配置为在仿真结束处输出所有节点的 IP 转发表。

注意最后两种方法仅适用于输出 IP 转发表。如果希望输出由一个特定路由协议产生的路由表，那么需要将 Reports... 属性设置为在所选择的节点处输出期望路由协议的路由表。例如，为了输出由 BGP 协议产生的路由表，需要在期望的节点处配置属性 **Reports... BGP Routing Table**（报告... BGP 路由表）。

所有这些方法将被选中节点的转发/路由表输出到扩展名为 .ot 的一个标准输出文件。通过 **Results Browser** 窗口中的 **DES Run Tables**，可在仿真完成时查看输出的表。如图 9.19 所示，分别通过单击 **Generate Web Report** 或 **Show** 按钮，可在一个编辑器中查看一个 Web 报告或查看表。注意默认 OPNET 编辑器允许用户将所显示的转发/路由表输出到一个电子表格、XML 或以逗号分隔的文本文件。

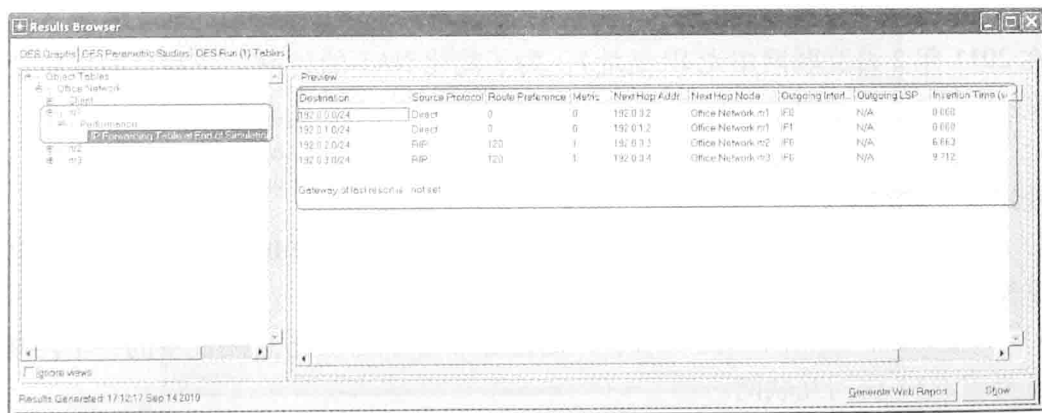


图 9.19　查看输出的转发表

通过设置 **IP Attribute Config** 工具节点中 **IP Route Table Export** 属性的值，用户也可将用户的仿真配置为输出所有节点的 IP 转发表到一个文本文件。**IP Route Table Export** 属性由属性 **Reports... IP Forwarding Table** 的相同子属性组成，因此可以类似方式配置。注意，虽然该属性被称作 **IP Route Table Export**，它实际上仅输出 IP 转发表。**IP Route Table Export** 属性也允许将仿真配置为不仅在仿真结束时输出 IP 转发表，而且在仿真过程中用户定义的时间输出 IP 转发表。所输出的信息被保存在默认目录中扩展名为 .gdf 的一个文本文件。典型情况下，输出的文件被命名为 < scenario name > _ < DES > - < run # > _ ip _ route _ tables _ < file # > . gdf。您可使用任何文本编辑器打开 .gdf 文件。

第10章　高级IP功能特征

本章描述在OPNET中支持的各种高级IP功能特征。本章伊始，描述配置网络地址转换（NAT）的可用OPNET选项，接下来描述IP组播和IPv6。通过给出在OPNET中建模的QoS机制的详细描述，结束本章。

10.1　网络地址转换（NAT）

10.1.1　NAT概述

网络地址转换（Network Address Translation，NAT）是将专用IP地址（如10.0.0.0/8、169.254.0.0/16、172.16.0.0/11和192.168.0.0/16）映射到全局唯一地址的一种机制。在NAT的帮助下，在具有有限数量全局IP地址的一个域中的所有主机，就可得到互联网的访问能力。开发这项技术是为了解决全局IP地址缩减数量的问题（在2011年2月互联网用光IP地址）。为了简化NAT的讨论，在本节后面将使用如下定义：

1）一个*private*（专用）或*local address*（本地地址）是一个未注册IP地址，不能官方地用在互联网中。

2）一个*global address*（全局地址）是一个全局唯一注册IP地址，可用于互联网中的数据交付。

3）一个*inside network*（内部网络）是包含带有专用IP地址的设备的一个网络。这些设备要求NAT访问互联网。

4）一个*outside network*（外部网络）是类似互联网的一个网络，包含带有全局IP地址的设备。

5）一个gateway（网关）是这样一个设备，它驻留在一个内部网络和一个外部网络之间的边界处。附接到一条链路（将一个网关分别连接到一个内部网络或外部网络）的一个接口称作一个内部接口或外部接口。

典型情况下，NAT被配置在内部网络和外部网络之间的一个网关或路由器节点内，如图10.1所示。NAT负责维护本地IP地址和全局IP地址之间的一个映射。存在几种类型的NAT，包括*static NAT*（静态NAT）（提供本地IP地址到全局IP地址的一对一转换）、*dynamic NAT*（动态NAT）（从一个可用全局地址池中为本地IP地址动态地选择一个转换匹配）、*port address translation*（端口地址转换）（PAT）（其中多个本地地址被转换为具有不同端口号的单个全局地址）以及其他类型。

典型情况下，NAT以如下方式工作：一个内部网络节点产生要发送到外部网络中



图 10.1　带有一台支持 NAT 的路由器的范例网络拓扑

一个目的地的一条请求。在将这条请求转发到外部网络之前，网关使用 NAT 以一个相应的全局地址和可能的一个端口号重写该请求报文的源 IP 地址和可能的目的端口 IP 首部字段。NAT 将本地和全局 IP 地址及端口号之间的映射记录在其转换表中。当响应从外部网络到达网关时，NAT 咨询其转换表，之后将响应报文的目的 IP 地址和可能的端口号改变为相应的本地 IP 地址和端口号。之后被更新的响应报文被转发进入内部网络。

在 OPNET 中，可在具有层 3 功能的核心节点（如路由器、网关、防火墙等）的模型中定义一个 NAT 配置，而边缘节点模型通常不支持 NAT 功能。通过设置一个节点的 **IP...NAT Parameters** 属性的值，可在该节点上配置 NAT 功能。如图 10.2 所示，属性 **NAT Parameters** 由如下子属性组成：

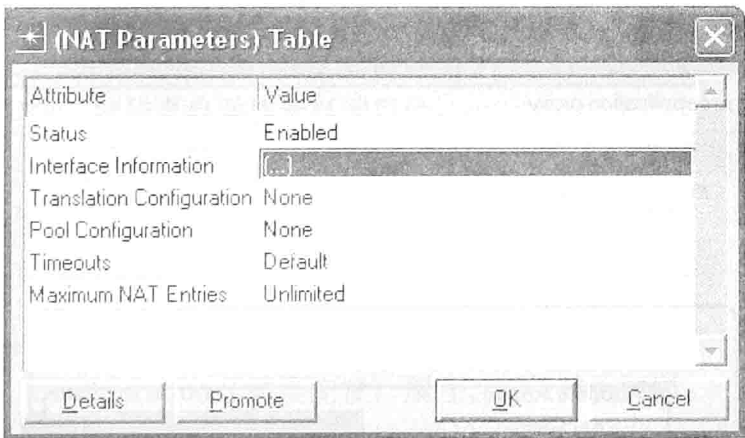


图 10.2　属性 **NAT Parameters**

1）**Status** 指定在这台路由器上是否激活 NAT 机制。

2）**Interface Information**（接口信息）识别和配置支持 NAT 功能的接口。典型情况下，配置 NAT 功能的过程，由指定部署于支持 NAT 接口的转换规则组成。

3）**Translation Configuration**（转换规则）定义 NAT 转换规则。

4）**Pool Configuration**（地址池配置）定义一个全局地址和可能的端口号集合。这个信息可用于本地地址的转换。

5）**Timeouts**（超时）为各种类型的流量指定 NAT 超时（即在清除一个空闲连接的转换信息之前，NAT 等待的时长）。

6）**Maximum NAT Entries**（最大 NAT 表项数）指定 NAT 转换表的最大尺寸。

10.1.2 配置 NAT

典型情况下，配置 NAT 的过程要求如下步骤：

1）在一台网关路由器中识别内部接口和外部接口的名字，由该网关实施 NAT。

2）如果需要，为转换规则指定地址池。

3）定义转换规则。

4）在步骤 1）中确定的接口上部署转换规则。

识别确定内部接口和外部接口名的第一步是非常直接的。它遵循 9.3.2 节中描述的相同过程，其中识别确定附接到一条链路的任意接口的名字，即在链路上“晃动”（hover）鼠标指针，检查发送器和接收器链路属性，或使用 **Edit Ports**（编辑接口）选项打开 **Select Port Assignment**（选择端口指派）窗口。在 10.1.3 ~ 10.1.5 节详细解释其他三个步骤。

10.1.3 指定地址池

如果转换规则不依赖于地址池，则可略去这个步骤。为了定义一个地址池，需要配置 **Pool Configuration**（地址池配置）属性，如图 10.3 所示。

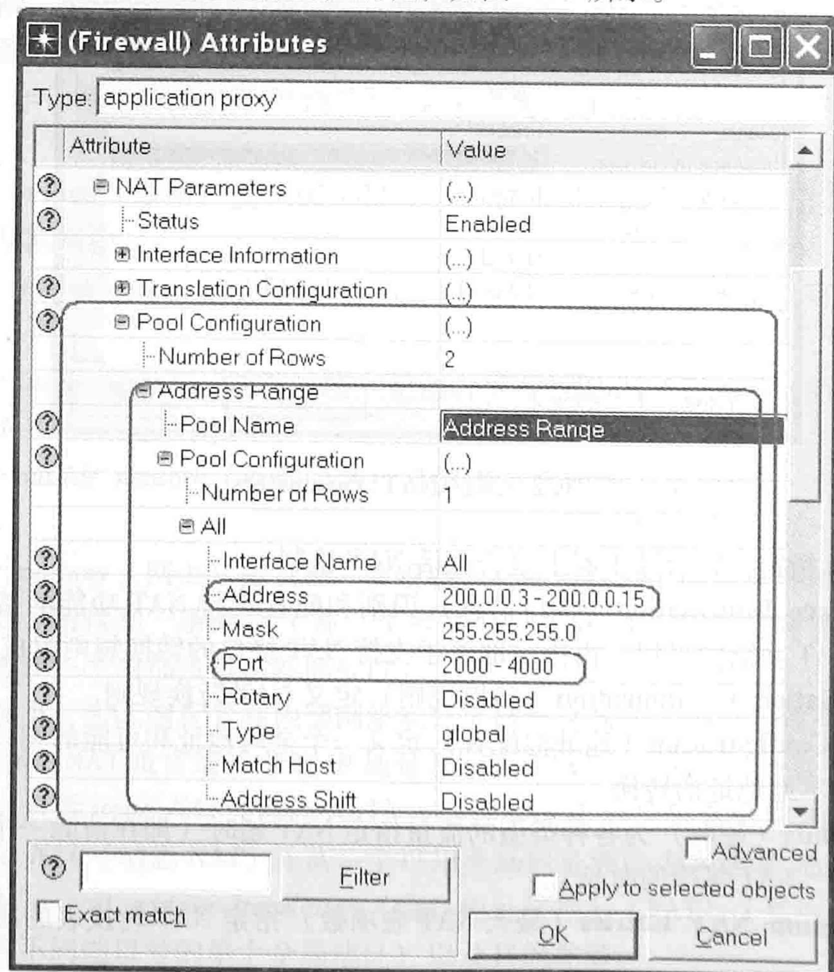


图 10.3 **Pool Configuration** 属性

首先，展开 **Pool Configuration** 属性，并设置属性 **Number of Rows** 的值，该属性指定要配置的地址池定义的总数。接下来，为每个新建的行实施如下步骤：

1）通过设置属性 **Pool Name**（池名）的值，指定地址池的名字。

2）设置属性 **Number of Rows** 的值，它对应于要在当前地址池内定义的不同地址范围的数量。

3）通过指定全局可路由的地址和可能的端口号，配置每个地址范围。下面是对定义地址范围的关键属性的描述。

① **Interface Name** 指定这个地址范围可被使用的外发接口的名字。可将这个属性值设置为 All（所有的），这使这个地址范围可用于所有外发接口。

② **Address** 为转换规则指定一个实际的 IP 地址或 IP 地址的一个范围。地址范围是使用如下表示法指定的：< start address > - < end address >（如 134. 13. 4. 0 ~ 134. 13. 4. 255）。

③ **Mask** 指定与 **Address** 属性一起使用的子网掩码。这个属性接受值 Not Assigned（未指派的），这将使仿真忽略掩码值。

④ **Port** 为当前地址范围指定一个端口号或端口号的一个范围。使用如下表示法指定端口范围：< start port number > - < end port number >（如 2000 - 3000）。属性 **Port** 也接受如下两个值：Not Assigned（未指派的），使仿真在转换过程中忽略端口号，以及 Automatic（自动地），使仿真自动地选择端口号。

⑤ **Type** 指定地址范围是为远端 VPN 客户端定义一个唯一全局地址还是本地地址，后者与专用地址是不同的。

10. 1. 4　指定转换规则

配置 NAT 的第三步是定义实际的转换规则。通过实施如下步骤，利用如图 10. 4 所示的 **Translation Configuration**（转换配置）属性，定义转换规则：

1）展开 **Translation Configuration** 属性，并设置属性 **Number of Rows** 的值，后者对应于要在这个节点上定义的 NAT 规则总数。

2）通过指定如下内容，配置每条 NAT 规则：

① 针对识别要由 NAT 转换的原报文的分类规则。

② 被转换报文的描述，即在 NAT 转换之后原报文看起来是什么样的。

③ 各种转换特征。

识别原报文的分类规则是通过 **Original Packets** 属性指定的，这允许针对到达报文的相应信息，如要匹配的源、目的地、协议、端口和应用信息。源和目的地信息可通过诸如 **IP address**（IP 地址）和 **Subnet Mask**（子网掩码）、**Access Control List**（访问控制列表）、**Route Map**（路由映射）和 **Network Object Group**（网络对象组）等属性加以指定。属性 **Protocol** 仅接受值 Any、TCP 或 UDP 等；属性 **Port** 允许指定实际的端口号；而属性 **Applications** 要求指定将匹配报文所属的应用的名字。

被转换报文特征的定义是通过属性 **Translated Packets** 指定的，由子属性 **Source**、

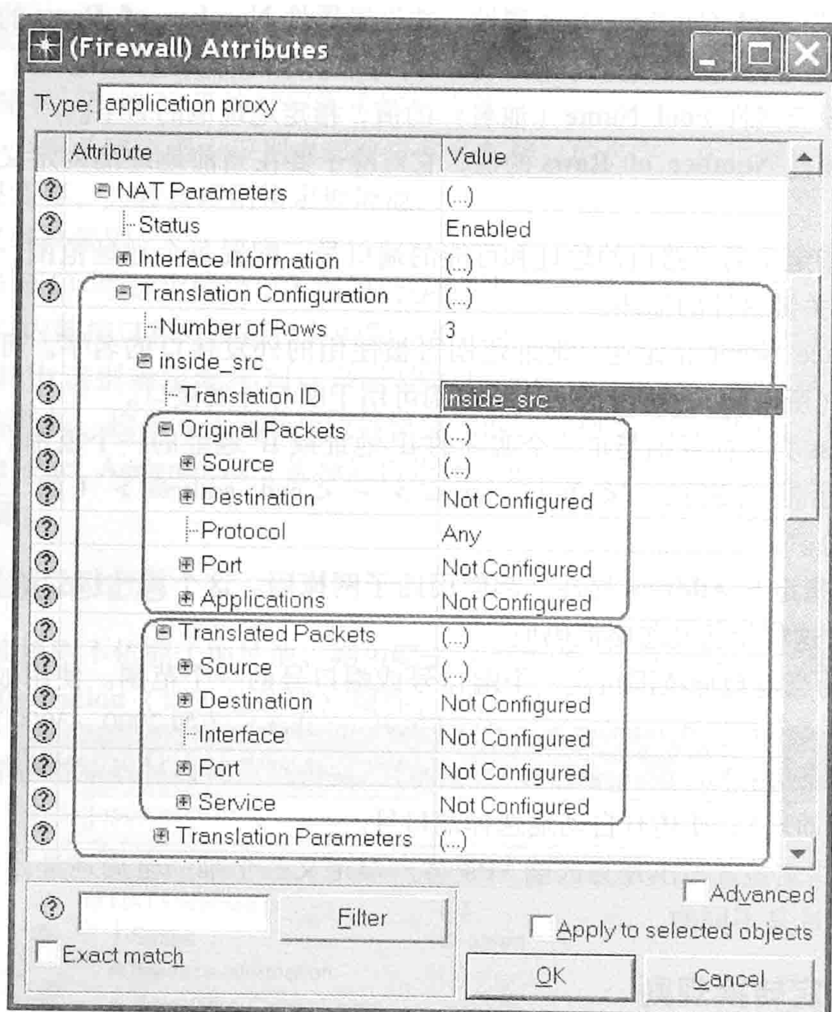


图 10.4　**Translation Configuration** 属性

Destination、**Interface**、**Port** 和 **Service** 组成。不需要为所有这些子属性指定值。在多数情形中，将需要设置源或目的地信息。**Source** 和 **Destination** 属性具有相同的结构，并仅通过如下属性之一定义源和目的地特征：

1）**IP Address**——被转换报文将使其源或目的地 IP 地址字段设置为指定的 IP 地址。

2）**Interface**——被转换报文将使其源或目的地 IP 地址设置为外发接口的地址。

3）**Pool**——这个属性仅接受已经定义的地址池的名字。被转换报文的源或目的地 IP 地址将从在这个字段中指定的地址池中进行选择。

4）**Network Object Group**（网络对象组）——将依据网络对象组的值，设置源或目的地信息，仅在一个防火墙节点上，网络对象组的值才是可配置的。在本书中不讨论这个功能特征。

如图 10.5 所示的属性 **Translation Parameters**（转换参数）指定如下信息：

1）转换模式类型［即 Dynamic（动态的）——从一个地址池中确定转换地址，

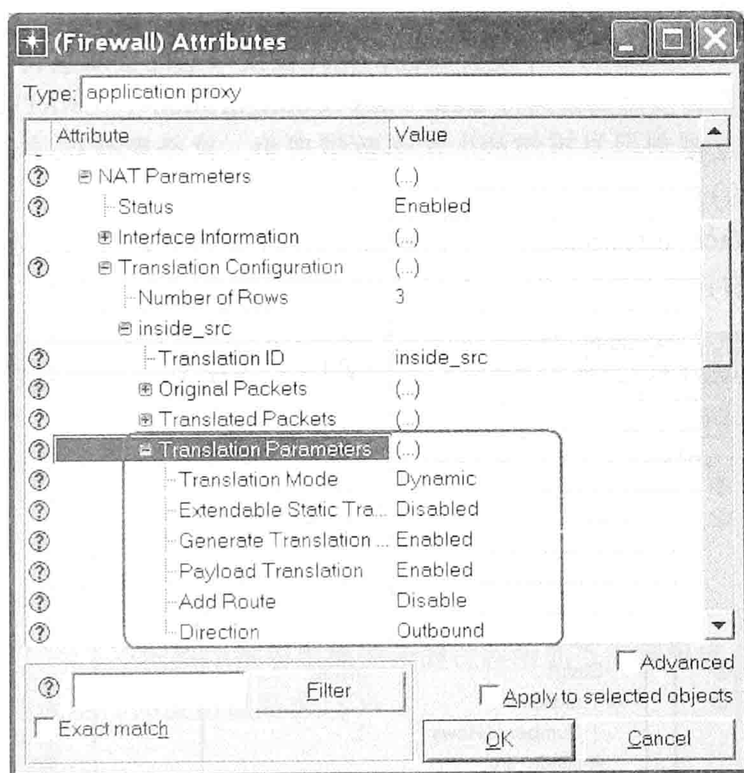


图10.5　**Translation Parameters** 属性

Static（静态的）——一对一地址转换，或 Overload（过载）——端口地址转换（PAT）]。

2）转换的方向［即 Inbound（入）或 Outbound（出），典型情况下为 Outbound］。

3）某些 NAT 特征［即 **Extendable Static Translation**（可扩展的静态转换）、**Generate Translation Alias**（产生转换别名）、**Payload Translation**（净荷转换）和 **Add Route**（添加路由）］是激活的还是禁止的。在多数情形中，这些属性仍然保持设置为其默认值。

10.1.5　在网关接口上部署转换规则

一旦定义转换规则，就可在10.1.2节第一步中确定的相应接口上部署这些规则。为部署转换规则，如图10.6所示，展开属性 **Interface Information**，之后设置属性 **Number of Rows** 的值，该值代表支持 NAT 的接口总数。每个新建的行代表单个接口的 NAT 配置。通过指定如下信息，需要配置每个单独的行：

1）**Name** 是要为 NAT 配置的接口名（如 IF0、IF5 等）。应该在过程的第一步确定这样的一个接口。

2）**Status** 指明当前接口将路由器连接到一个内部网络还是外部网络。这个属性仅适用于 Cisco IOS NAT，在 Cisco PIX 防火墙设备中被忽略。

3）**Translations** 为一个复合属性，指定部署在这个接口上的转换规则。可配置一

个接口，支持在 **Translation Configuration** 属性中定义的多条规则。

4）**Dual Translations** 为一个复合属性，允许在接口上配置双重转换（dual－translation）规则。双重转换规则是通过 **Dual NAT Configuration**（双 NAT 配置）属性指定的，仅在某些节点模型中是可用的。在多数情形中，这个属性保持设置为 Not Configured（未配置的）。

5）**Subinterface Information**（子网信息）允许在节点的子接口上指定 NAT。注意，如果节点没有定义好的子接口，则不能配置这个属性。

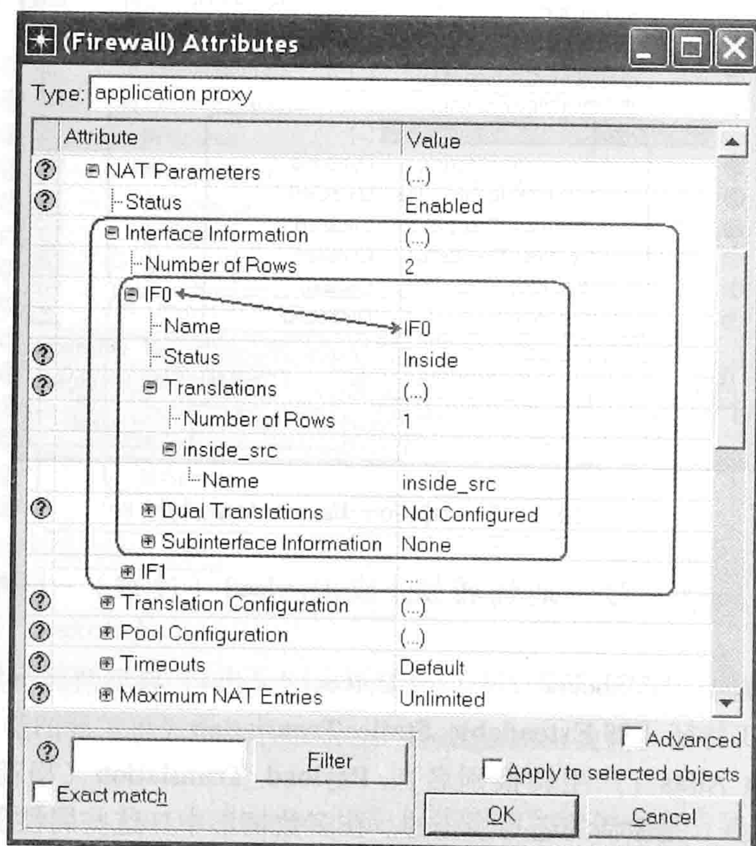


图 10.6 **Interface Information** 属性

作为标准 OPNET 软件发布组成部分的项目 IP，包含称为 *NAT* 的一个场景，它提供了在一个网络中 NAT 配置的绝佳范例。

10.2 IP 组播

10.2.1 在 OPNET 中支持的 IP 组播功能特征

IP 组播是允许一对多通信的一项技术，即单个源将数据传输到多个目的地。OPNET Modeler 和 IT Guru 支持 IP 组播，并允许网络模型的开发，其中包括互联网组管理协议（IGMP）和协议无关组播—稀疏模式（PIM－SM）。另外，OPNET 的 IP 组播模型

允许使用如下方法指定一个聚合点（RP）：

1）*Static - RP*（静态 RP）——所有组播路由器都显式配置有 RP 信息。

2）*Auto - RP*（自动 RP）——仅有某些组播路由器配置有 RP 信息。其他路由器在仿真过程中动态地发现 RP 信息。自动 RP 方法依赖于映射代理将 RP 信息分布在被仿真网络中。

3）*Bootstrap*（启动）——同样不需要所有路由器静态地配置 RP 信息。相反，一台 Bootstrap 路由器（BSR）负责将 RP 信息分布在网络中的其他路由器间。

当前，可使用某些运行在 UDP 之上的标准应用、运行在 UDP 之上的单向定制应用和 IP 组播需求（即需求对象 *ip _ mcast _ traffic _flow*），定义 IP 组播流量。特别地，可仅针对话音、视频、数据库和 FTP 标准应用指定组播流量。对于话音和视频应用，仅有工作站节点可作为组播流量的一个目的地。对于其他支持的标准应用，仅有服务器节点可接收组播流量。

10.2.2　部署 IP 组播流量的步骤概述

典型情况下，在一个网络中部署组播流量的过程由如下步骤组成：

1）定义组播流量（即应用或需求）。

2）配置源节点。

① 部署定义好的组播流量（仅适用于使用标准应用的情况）。

② 在接口上激活组播。

3）配置目的地节点。

① 激活对组播应用服务的支持（仅适用于使用标准应用的情况）。

② 为加入/离开一个组播目的地组指定细节。

③ 在接口上激活组播。

4）配置组播路由器。

① 在将承载组播流量的接口上激活组播。

② 使用静态 RP、自动 RP 或启动机制指定 RP。

将在 10.2.3 ~ 10.2.7 节详细讨论这些步骤。但是，IP 组播是一个十分复杂的话题，这里仅描述部署 IP 组播的关键配置步骤。欲了解有关 IP 组播的更多信息，参见 RFC 1112、2236、2362 和 4601 或其他外部来源。OPNET 文档的 STM - 13 一章是有关 OPNET 支持 IP 组播的一个绝佳信息源。另外，Modeler 和 IT Guru 包含一个名为 IP _ Multicast 的一个预配置样例项目，它包含几个场景，对在 OPNET 中的 IP 组播部署和配置做了绝好展示。

10.2.3　定义组播流量

与任何其他标准一样，可为 IP 组播定义用户概要和标准应用。因为这个应用流量的目的地是一个组播组，所以建议将 **Symbolic Destination Name**（符号目的地名）属性的值设置为指明这个事实的一个名字。例如，在 **IP _ Multicast** 项目的 **Multicast** 场景

中，组播话音应用的 **Symbolic Destination Name** 属性的值设置为 Multicast Receiver。而且，应该确保组播应用运行在 UDP 之上。回顾一下，在高级工作站和服务器模型中，通过属性 **Applications... Application: Transport Protocol Specification**（各项应用...应用：传输协议规范），可控制由一个应用使用的传输协议类型。

添加一个 IP 组播需求对象的过程是稍稍有点复杂的，且由如下步骤组成：

1）在 **Object Palette Tree** 中的 *ip _ mcast _ traffic _flow* 对象上双击。可在 *demands* 类中找到这个对象。

2）在将作为组播流量之源的节点上点击。但是，不要将组播需求对象连接到任何目的地节点。

3）在项目工作空间的空区域的任何地方双击，这将一个组播需求对象添加到被选中的节点。

4）可重复这个过程并添加其他需求对象，或可终止添加对象，方法是在项目工作空间的任何地方右击，并从出现的菜单中选择 **Abort Demand Definition**（中止需求定义）选项。

一旦添加了所有的组播需求对象，则就需要配置每个对象，方法是至少指定目的地 IP 地址和流量速率。图 10.7 给出一个 IP 组播需求对象的配置属性。通过设置属性 **Destination IP Address** 的值，可指定组播组地址。所提供的地址值必须对应于一个有效的组播地址（如范围 224.0.0.0/4 中的任意地址）。组播需求流量速率是通过属性 **Traffic（bits/second）**［流量（比特/秒）］和 **Traffic（packets/second）**［流量（报文数/秒）］指定的。也可能希望设置属性 **Traffic Start Time**（流量开始时间）的值，该属性控制需求开始产生流量的时间。默认条件下，这个属性被设置为 Same As Global Setting（与全局设置相同）［即通过 **Config/Run DES** 窗口中的全局属性 **Traffic... Background Traffic Start Delay（seconds）** 定义属性值］。注意，如果目的地节点在需求开始流量传输之后加入组播组，那么在目的地加入组播组之前发送的所有报文都将被丢弃。典型情况下，可容易地识别这样一个问题，原因是在出现一个组播需求报文被丢弃（由于组播组中没有目的地）的事件下，DES 日志记录一条消息。

10.2.4 配置源节点

当配置源节点时，第一步是在节点上激活 IP 组播。通过如下步骤之一，可在期望的源节点上激活 IP 组播：

1）将 **IP... IP Host Parameters... Multicast Mode**（IP... IP 主机参数...组播模式）属性的值设置为 Enabled。

2）选择希望其作为组播源的所有节点，并从 **Project Editor** 的下拉菜单中选择选项 **Protocols→IP→Multicast→Enable Multicasting on Selected Hosts**（协议→IP→组播→在被选中主机上激活组播），接下来将在所选中节点上将 **IP... IP Host Parameters... Multicast Mode**（IP... IP 主机参数...组播模式）属性的值改变为 Enabled。

如果通过需求对象部署了所有的组播流量，那么这一步就完成了源节点配置。但

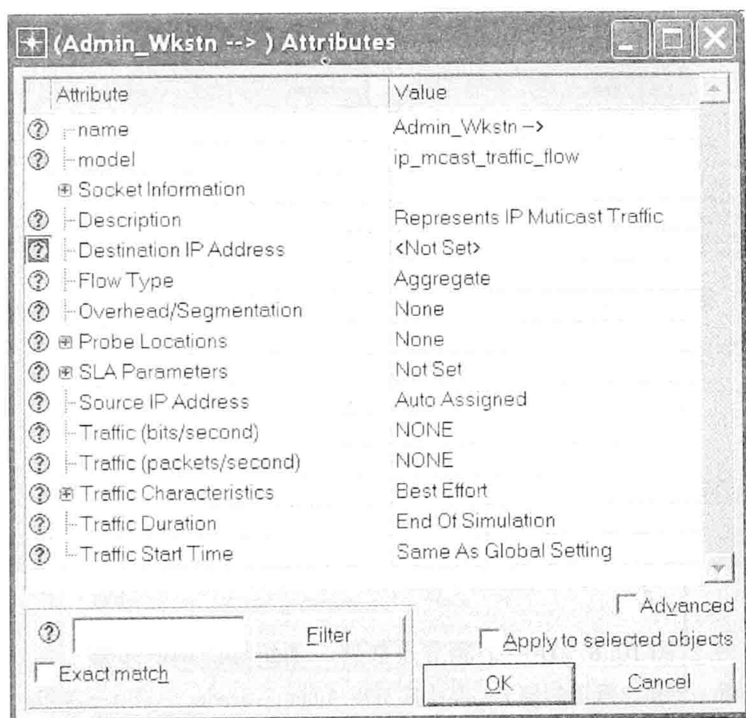


图 10.7　一个 IP 组播需求对象的配置属性

是，如果 IP 组播流量是通过定制或标准应用定义的，那么需要部署这样的概要，这些概要将在期望的源节点上运行所定义的组播应用。OPNET 不支持定制应用的自动部署，这意味着不能使用 **Protocols→Applications→Deploy Defined Applications...**（协议→应用→部署定义好的应用...）选项来部署组播应用。因此，为了部署组播应用，需要手工配置源和目的地节点。

为了将一个节点定义为一项组播应用的一个源，实施如图 10.8 所示的如下两步：

1）通过设置属性 **Applications...Application：Supported Profiles**（各项应用...应用所支持的概要），配置节点支持一个组播应用概要。

2）通过将应用的符号服务器名映射到组播组地址，为组播应用定义目的地首选项。回顾一下，通过设置属性 **Applications...Application：Destination Preferences**（各项应用...应用目的地首选项），可配置这样的一个映射。注意，实际的服务器地址（即属性 **Name**）必须被设置为一个有效的组播地址。

如图 10.8a 所示，节点 Admin _ Sender 被配置为支持称为 Video 的一个概要，它包括带有符号服务器名设置为 Multicast Receiver 的一项组播应用。图 10.8b 表明，节点 Admin _ Sender 将符号服务器名 Multicast Receiver 映射到组播地址 224.0.6.1，该地址实际上定义组播组地址。

10.2.5　配置目的地节点

在开始配置组播目的地节点之前，需要确保目的地节点是通过中间节点模型或高级

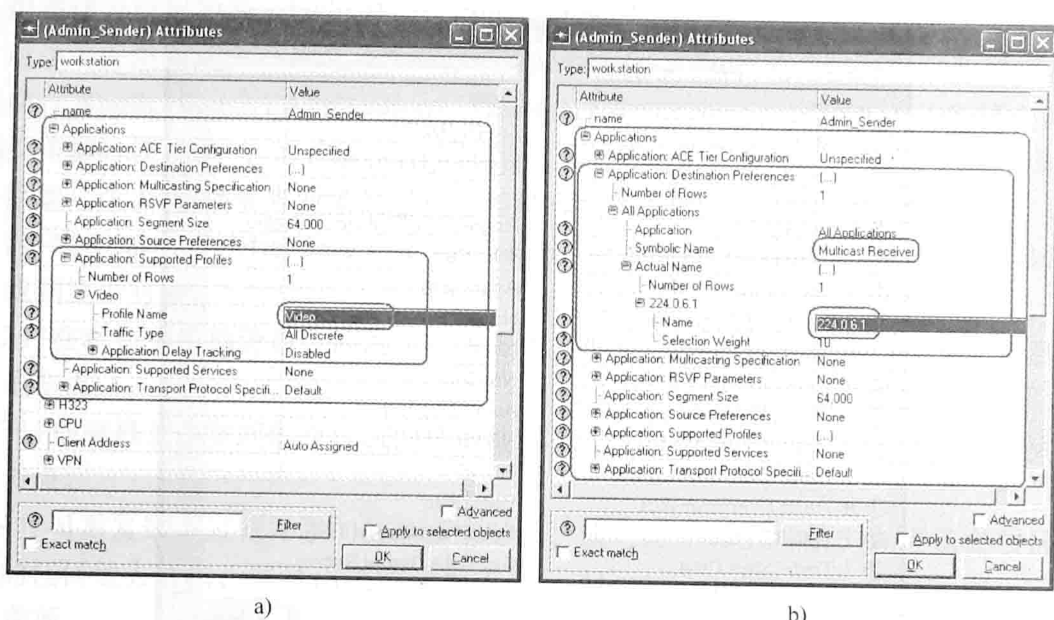


图 10.8　在一个源节点部署一项组播应用的步骤

a）部署一个用户概要的例子，其中在节点 Admin _ Sender 中带有一项组播应用

b）将符号名 Multicast Receiver 映射到一个特定组播组地址的例子

节点模型表示的（即模型名有_ *adv* 或_ *int* 修饰符）。否则，对配置 IP 组播所必要的某些功能特征将是不可用的。典型情况下，配置目的地节点的过程由如下步骤组成：

1）在目的地节点上指定组播组。

2）在目的地节点上激活组播。

3）配置目的地节点支持组播应用服务。仅当组播流量是使用标准或定制应用加以指定时，这个步骤才可适用。

通过如下方法之一，可指定一个目的地节点是一个组播组的组成部分：

1）对于属于组播组的每个目的地节点，通过至少设置如下属性的值，配置属性 **Applications... Application：Multicasting Specification**（各项应用... 应用：组播规范）：

① **Application Name**（应用名）——组播应用的名字。

② **Membership Address**（成员关系地址）——组播组地址，即映射到组播应用的符号服务器名的 IP 地址值，或在 IP 组播需求对象的 **Destination IP Address** 属性中设置的。

③ **Join Time（seconds）**［加入时间（秒）］——当节点加入组播组时的时间。

④ **Leave Time（seconds）**［离开时间（秒）］——当节点离开组播组时的时间。

在此之后，通过将属性 **IP... IP Host Parameters... Multicast Mode**（IP... IP 主机参数... 组播模式）的值设置为 Enabled，也需要在目的地节点上激活组播。图 10.9 展示说明了 **Application：Multicasting Specification** 属性。注意这个属性仅在高级端节点模型中才是可用的。

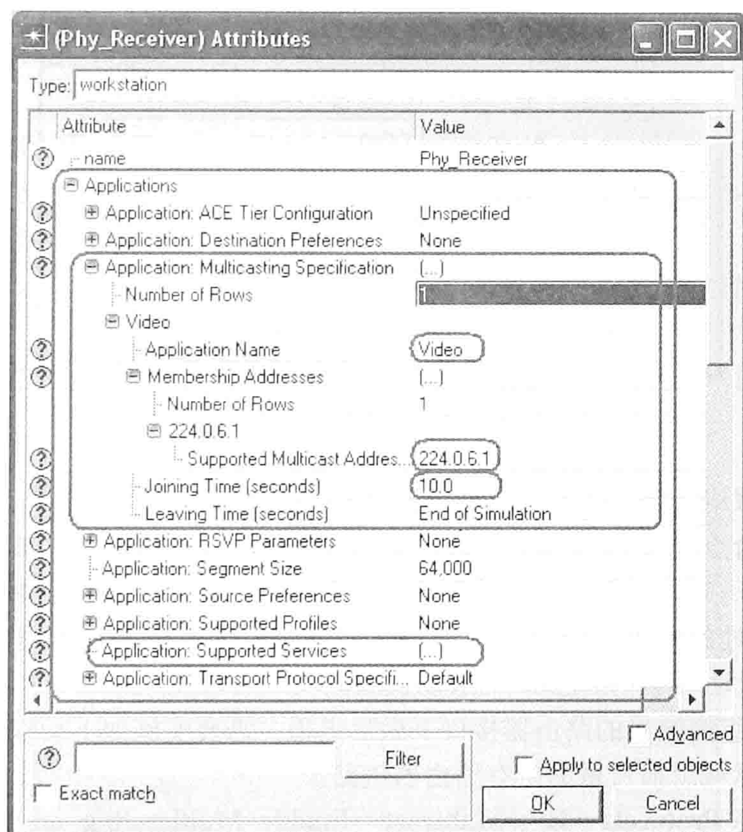


图 10.9　**Application：Multicasting Specification** 属性

2）采用另外的方法也可取得相同效果，方法是通过选择组播目的地节点，之后使用 **Protocols→IP→Multicast→Set Multicast Group on Selected Destination Nodes**（协议→IP→组播→在被选中的目的地节点上设置组播组）选项，打开 **Destination nodes** 窗口，如图 10.10 所示。注意，除非被选中的节点都使用高级端节点模型，否则 **Destination nodes**（目的地节点）窗口中的所有字段都将是不被激活的。同样，需要为 **Application**（应用）、**IP Multicast Group Address**（IP 组播组地址）、**Join Time（seconds）**［加入时间（秒）］和 **Leave Time（seconds）**［离开时间（秒）］属性指定值。通过单击 **OK** 按钮，可保存所引入的改变。注意，得到的系统配置将与第一种方法中实施所描述步骤的结果相同，即所有被选中的节点将使属性 **Multicast Mode**（组播模式）设置为 Enabled，依据 **Destination nodes** 窗口中指定的设置配置属性 **Applications... Application：Multicasting Specification**。

最后，如果通过标准或定制应用来指定组播流量，那么需要指定每个目的地节点支持组播应用的服务，方法是配置节点的属性 **Applications... Application：Supported Services**（各项应用... 应用：支持的服务），该属性在图 10.9 的底部以圆圈形式标出。

10.2.6　配置路由器节点

配置核心路由器转发组播流量，是一个两步骤的过程：

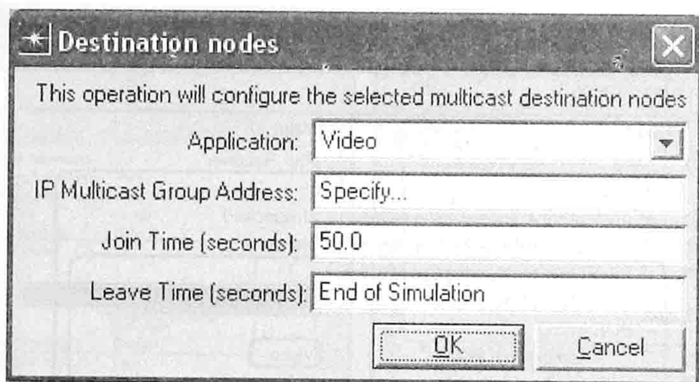


图 10.10　**Destination nodes** 窗口

1）在将承载组播流量的接口上激活组播。

2）指定聚合点（RP）。RP 经常被定义为网络中的一个点，其中来自源和接收方的组播数据聚合在一起。实际上，它是网络中的一个点，其中组播数据源自一个源，在组播数据到各目的地的路上，开始在一个共享的分布树的不同路径上进行分布。OPNET 支持静态 RP、自动 RP 和指定 RP 的启动机制。

为将在承载组播流量的路由器接口上激活组播，需要实施如下动作：

1）选择将承载组播流量的所有路由器链路。

2）通过选择 **Protocols→IP→Multicast→Enable Multicasting on Interfaces across Selected Links**（协议→IP→组播→在选中链路间的接口上激活组播）选项，在选中的链路间激活组播。作为这项操作的结果，将在附接于选中的链路的所有路由器接口上激活 IP 组播。

指定 RP 的过程的第一步，是确定将作为组播网络的分发树中 RP 的路由器。具体而言，需要在 RP 路由器上找到一个活跃的支持组播接口的 IP 地址。我们将称这样一个值为 *RP address*（RP 地址）。如果在网络中还没有配置 IP 地址，那么可在相应的路由器接口上手工输入 IP 地址，或像建议的那样，使用 **Protocols→IP→Addressing→Auto－Assign IPv4 Addresses**（协议→IP→编址→自动指派 IPv4 地址）选项，在网络的所有节点中指定 IP 地址。一旦指派了 IP 地址，就需要检查 **IP... IP Routing Parameters... Interface Information**（IP... IP 路由参数... 接口信息）属性确定 RP 地址。在 RP 路由器中任意活跃的支持组播的接口的 IP 地址可作为 RP 地址。

现在您准备好在被仿真系统中指定 RP。取决于选择的 RP 机制的类型，配置步骤是不同的。我们顺次描述每种机制的配置步骤。注意，通过在个体节点处手工地改变属性值，可配置组播路由器。但是，手工在路由器接口上激活组播，之后手工定义 RP，是一项烦人的和容易出错的任务，这就是强烈鼓励依靠本节中描述的 **Protocols** 菜单选项的原因。

1. 静态 RP 机制

使用静态 RP 机制，执行定义 RP 的如下步骤：

1）选择转发组播流量的路由器。如果希望在网络的所有路由器上指定 RP，那么就

可忽略这一步。

2）选择 **Protocols→IP→Multicast→Configure Rendezvous Point Using Static RP Configuration...**（协议→IP→组播→配置聚合点使用静态 RP 配置...）选项，打开如图 10.11 所示的窗口。

3）设置 *IP Multicast Group Address/Mask*（IP 组播组地址/掩码）字段的值。注意，所输入的值必须是以 CIDR 表示法写出的一个有效组播地址。

4）将 *Rendezvous Point IP Address*（聚合点 IP 地址）字段的值设置为前面确定好的 RP 地址。

5）选择您的配置是应用到网络中的所有路由器还是仅应用到选中的路由器。

6）单击 **OK** 按钮保存所做的改变，并定义 RP 使用静态 RP 机制。

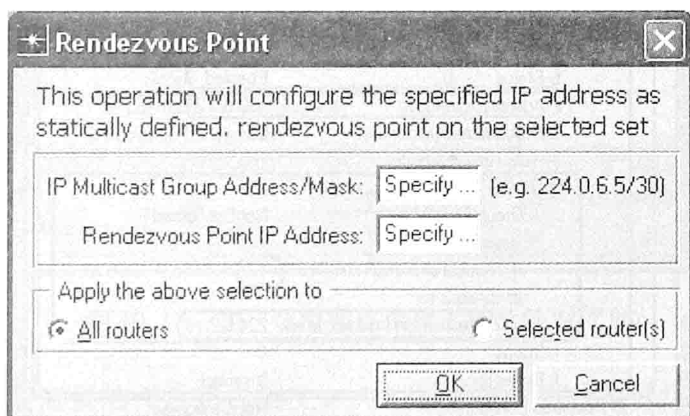


图 10.11　配置 RP 使用静态 RP 机制

2. 自动 RP 机制

在静态 RP 机制中，每台路由器在仿真之前都被静态地指定 RP，与此不同，自动 RP 机制在仿真过程中动态地分发 RP 信息。为配置自动 RP 机制（是 Cisco 的专用机制），需要执行如下步骤：

1）在承载组播流量的任意路由器上，通过实施如下动作，为自动 RP 机制指定 RP 候选。这些步骤如图 10.12 所示。

① 展开属性 **IP Multicasting... PIM Parameters... Auto – RP Configuration**（IP 组播...PIM 参数... 自动 RP 配置）。

② 将属性 **Discover**（发现）设置为 Enabled。

③ 展开属性 **Candidate RP Configuration**（候选 RP 配置），并将属性 **Number of Rows** 的值设置为 1。

④ 展开新建的行，并将属性 **Status** 设置为 Enabled，将属性 **Address/Interface**（地址/接口）设置为前面确定的 RP 地址。

⑤ 展开属性 Group Filter Configuration... Groups（组过滤器配置... 组）。

⑥ 将属性 **Number of Rows** 的值［即 **Groups**（组）的子属性］设置为 1。

⑦ 展开新建的行，并将 **Destination Address/Mask**（目的地地址/掩码）属性的值

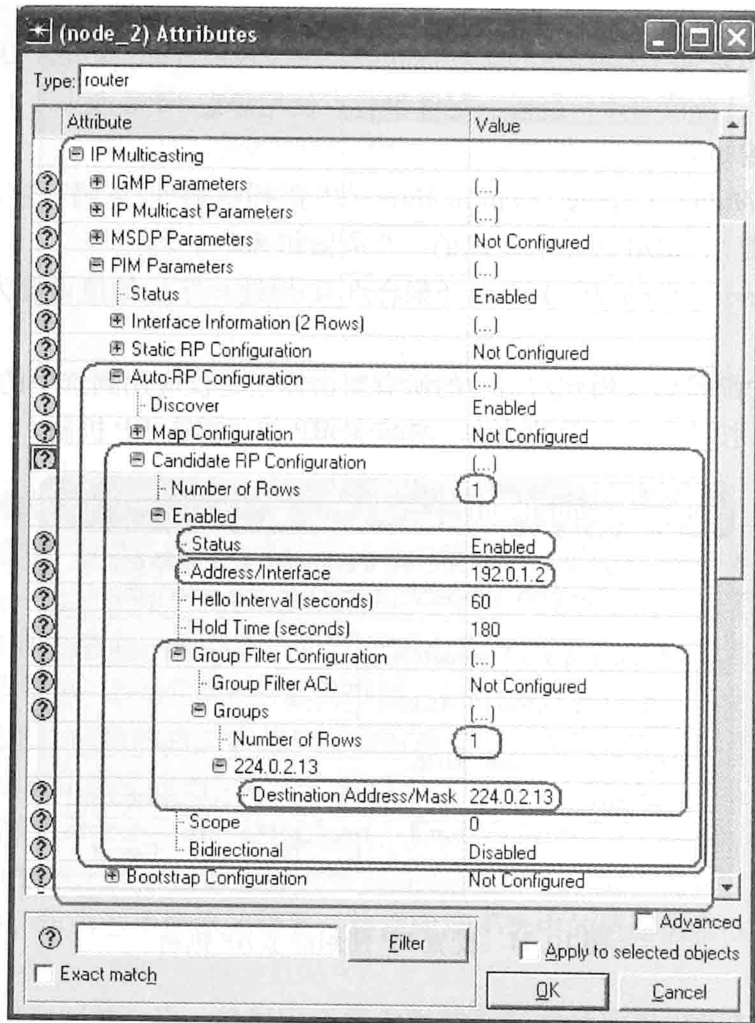


图 10.12 为自动 RP 配置机制指定一个 RP 候选

设置为以 CIDR 表示法写出的组播组地址。

⑧ 单击 OK 按钮保存所做的改变。

2）在网络中的组播路由器上激活自动 RP 机制，方法是首先选择所有的组播路由器，之后选择 **Protocols→IP→Multicast→Enable Auto – RP on Selected Routers**（协议→IP→组播→在选中的路由器上激活自动 RP）。

3）在至少一台组播路由器上激活映射代理，方法是将属性 **IP Multicasting... PIM Parameters... Auto – RP Configuration... Map Configuration... Status**（IP 组播... PIM 参数... 自动 RP 配置... 映射配置... 状态）设置为 Enabled，如图 10.13 所示。

3. 启动机制

启动或启动路由器（BSR）机制也动态地分发 RP 信息，总体而言非常类似于自动 RP 机制，另外情况是几个实现细节，这超出了本书的范围。BSR 是一种业界标准机制。为了配置启动机制，需要执行如下步骤：

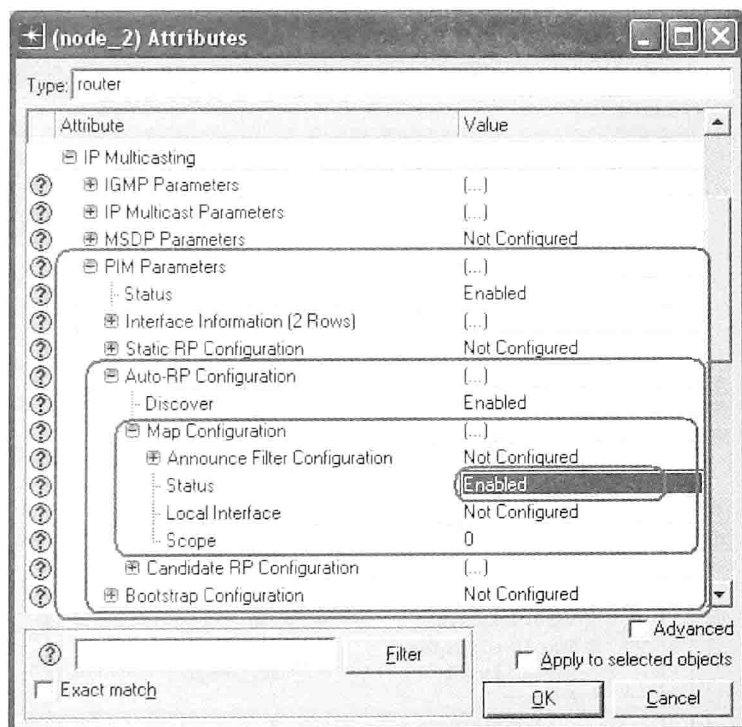


图 10.13　在一台组播路由器上激活映射代理

1）在承载组播流量的任意路由器上，为启动配置指定 RP 候选，方法是实施如下动作，如图 10.14 所示：

① 展开属性 **IP Multicasting... PIM Parameters... Bootstrap Configuration... Candidate RP Configuration**（IP 组播 ... PIM 参数 ... 启动配置 ... 候选 RP 配置），并将属性 **Number of Rows** 的值设置为 1。

② 展开新建的行，并将属性 **Status** 设置为 Enabled，将属性 **Address/Interface** 设置为前面确定的 RP 地址。

③ 展开属性 **Group Filter Configuration... Groups**（组过滤器配置 ... 组）。

④ 将属性 **Number of Rows**（即 **Groups** 的子属性）的值设置为 1。

⑤ 展开新建的行，并将 **Destination Address/Mask** 属性的值设置为以 CIDR 表示法写出的组播组地址。

⑥ 单击 **OK** 按钮保存所做的改变。

2）通过实施如图 10.15 所示的如下任务，选择组播路由器之一作为一个候选 BSR：

① 展开属性 **IP Multicasting... PIM Parameters... Bootstrap Configuration... Candidate BSR Configuration**（IP 组播 ... PIM 参数 ... 启动配置 ... 候选 BSR 配置）。

② 将属性 **Local Interface**（本地接口）的值设置为 Auto Assign（自动指派）或设置为这台路由器的接口（将被通告作为一个候选 BSR）的一个 IP 地址。

③ 将属性 **Priority**（优先级）设置为大于 0 的一个值。这个属性代表启动候选的优

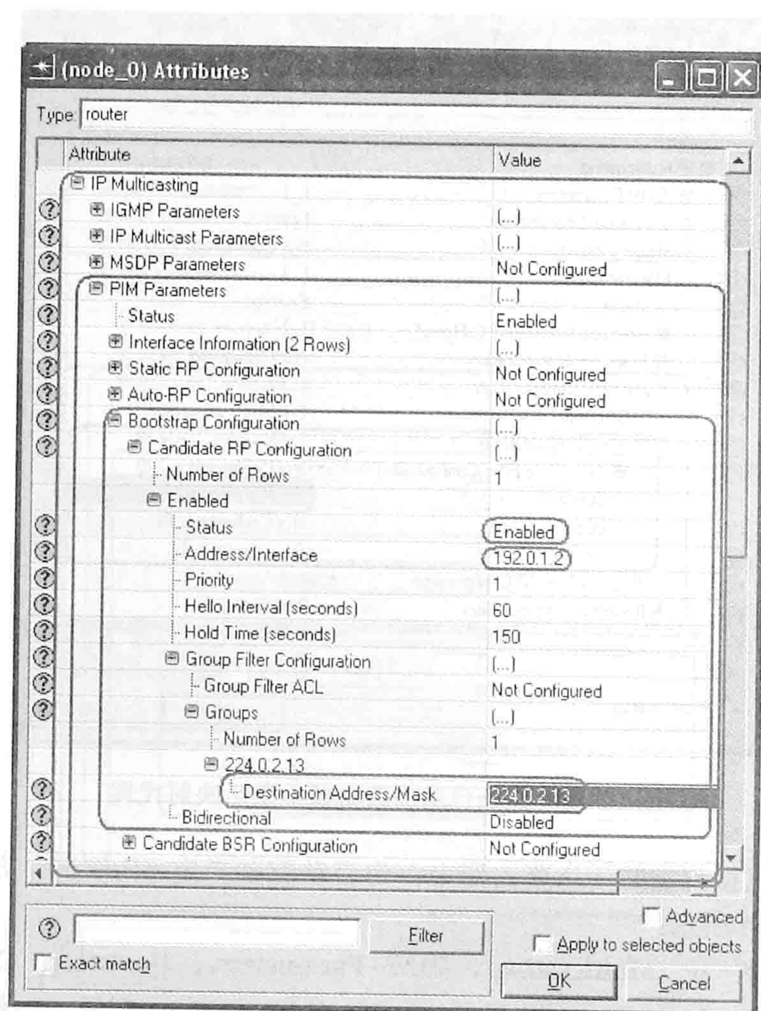


图 10.14 为启动配置指定一个 RP 候选

先级。具有最高优先级值的路由器被选举为 BSR。如果多台路由器有最高优先级值，那么具有最高 IP 地址值的路由器被选作 BSR。

10.2.7 其他组播配置参数

在 10.2.2 ~ 10.2.6 节，描述在一个被仿真网络中部署组播流量的基本过程，并简短地描述了一些可用属性。OPNET 软件包含了配置 IP 组播以及有关功能特征的许多其他属性，其中包括 IGMP 版本 1 ~ 3、源特定的组播、组播 VPN 等。为了了解有关所支持的组播功能特征的更多信息，请通过 **Protocols→IP→Multicast→Model User Guide**（协议→IP→组播→模型用户指南）访问产品文档。

10.2.8 组播统计和报告

OPNET 为评估组播流量和相关协议的性能，提供了一个统计集合。如图 10.16 所示，组播特定的节点统计位于 **IGMP Host**（IGMP 主机）、**IGMP Router**（IGMP 路由

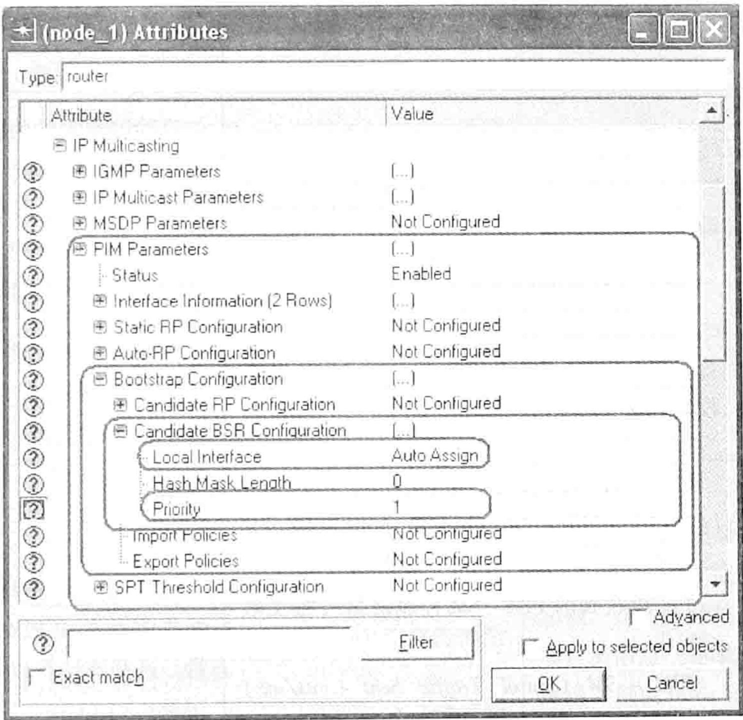


图 10.15　为启动机制指定一个候选启动路由器

器）、**IP**、**IP Interface**（IP 接口）、**IPv6** 和 **PIM – SM** 统计类，而仅有 PIM – SM 类包含

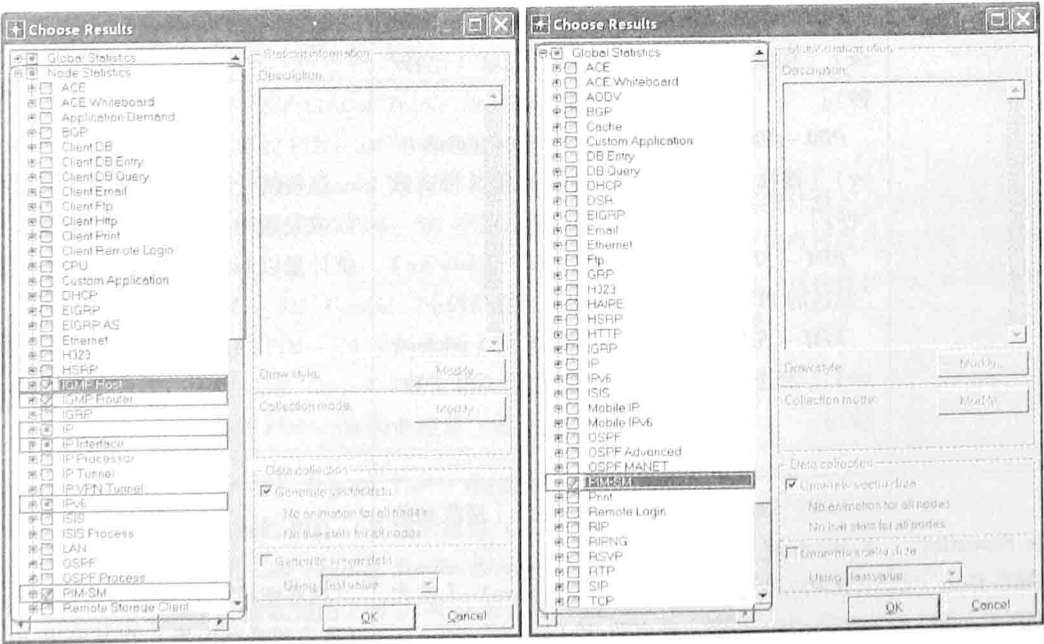


图 10.16　组播特定的统计类

组播特定的全局统计量。注意，**IP**、**IP Interface** 和 **IPv6** 类包含许多 IP 相关的统计量，但仅有其中一些是与组播相关的。在表 10.1 中给出组播特定统计量的一个汇总。

表 10.1　IP 组播统计量汇总

分　类	名　字	描　述
Global Statistics PIM – SM （全局统计量 PIM – SM）	*Network Convergence Activity*（网络收敛活动）	当检测到收敛的信号时的时段，被记录为 1。没有收敛活动的时段被记录为 0。得到的图由 1 和 0 的时段组成，在整个网络中对应于有收敛活动和没有收敛活动的时段
	Network Convergence Duration（*sec*）［网络收敛时长（秒）］	这个统计量在整个网络间记录 PIM – SM 路由表之收敛周期的时长。为每个组播组记录一个独立的统计向量
	PIM – SM Control Traffic Received（*bits/sec*）［接收到的 PIM – SM 控制流量（比特/秒）］ *PIM – SM Control Traffic Received*（*packets/sec*）［接收到的 PIM – SM 控制流量（报文数/秒）］ *PIM – SM Control Traffic Sent*（*bits/sec*）［发送的 PIM – SM 控制流量（比特/秒）］ *PIM – SM Control Traffic Sent*（*packets/sec*）［发送的 PIM – SM 控制流量（报文数/秒）］	这些统计量记录由网络中所有节点接收或发送的 PIM – SM 控制流量总量。这些统计量以 bit/s 和报文数/秒记录
	PIM – SIM Register Traffic Received（*bits/sec*）［接收到的 PIM – SIM 注册流量（比特/秒）］ *PIM – SIM Register Traffic Received*（*packets/sec*）［接收到的 PIM – SIM 注册流量（报文数/秒）］ *PIM – SIM Register Traffic Sent*（*bits/sec*）［发送的 PIM – SIM 注册流量（比特/秒）］ *PIM – SIM Register Traffic Sent*（*packets/sec*）［发送的 PIM – SIM 注册流量（报文数/秒）］	这些统计量记录网络中所有节点接收或发送的 PIM – SM 注册总量。这些统计量以 bit/s 和报文数/s 记录
Node Statistics IGMP Host IGMP Router （节点统计量 IGMP 主机 IGMP 路由器）	*IGMP Traffic Received*（*bits/sec*）［接收到的 IGMP 流量（比特/秒）］ *IGMP Traffic Received*（*packets/sec*）［接收到的 IGMP 流量（报文数/秒）］ *IGMP Traffic Sent*（*bits/sec*）［发送的 IGMP 流量（比特/秒）］ *IGMP Traffic Sent*（*packets/sec*）［发送的 IGMP 流量（报文数/秒）］	这些统计量记录网络中由这个节点（即主机或路由器）在其所有 IP 接口上接收或发送的 IGMP 消息总数。这些统计量以 bit/s 和报文数/秒记录

（续）

分　类	名　字	描　述
Node Statistics （节点统计量） **IP**	*Multicast Traffic Dropped*（*bits/sec*）［丢弃的组播流量（比特/秒）］ *Multicast Traffic Dropped*（*packets/sec*）［丢弃的组播流量（报文数/秒）］	这些统计量记录这个节点在其所有 IP 接口上丢弃的 IP 组播报文总数。这些统计量以 bit/s 和报文数/s 记录
	Multicast Traffic Received（*bits/sec*）［接收到的组播流量（报文数/秒）］ *Multicast Traffic Sent*（*packets/sec*）［发送的组播流量（报文数/秒）］	这些统计量记录这个节点在其所有 IP 接口上接收或发送的 IP 组播报文总数
Node Statistics **IP Interface** （节点统计量 IP 接口）	*Multicast Traffic Received*（*bits/sec*）［接收到的组播流量（比特/秒）］ *Multicast Traffic Received*（*packets/sec*）［接收到的组播流量（报文数/秒）］ *Multicast Traffic Sent*（*bits/sec*）［发送的组播流量（比特数/秒）］ *Multicast Traffic Sent*（*packets/sec*）［发送的组播流量（报文数/秒）］	这些统计量记录在一个特定接口上由这个节点接收或发送的 IP 组播报文总数。这些统计量记录具有组播地址和注册消息（承载数据）的报文。这些统计量以 bit/s 和报文数/s 记录
Node Statistics **IPv6** （节点统计量 IPv6）	*Multicast Traffic Received*（*packets/sec*）［接收到的组播流量（报文数/秒）］ *Multicast Traffic Sent*（*packets/sec*）［发送的组播流量（报文数/秒）］	这些统计量记录这个节点在其所有 IP 接口上接收或发送的 IPv6 组播报文总数
Node Statistics **PIM－SM** （节点统计量 PIM－SM）	*PIM－SM Control Traffic Received*（*bits/sec*）［接收到的 PIM－SM 控制流量（比特/秒）］ *PIM－SM Control Traffic Received*（*packets/sec*）［接收到的 PIM－SM 控制流量（报文数/秒）］ *PIM－SM Control Traffic Sent*（*bits/sec*）［发送的 PIM－SM 控制流量（比特/秒）］ *PIM－SM Control Traffic Sent*（*packets/sec*）［发送的 PIM－SM 控制流量（报文数/秒）］	这些统计量记录这个节点在其所有接口上接收或发送的 PIM－SM 控制流量总量。这些统计量以 bit/s 和报文数/s 记录
	PIM－SM Register Traffic Received（*bits/sec*）［接收到的 PIM－SM 注册流量（比特/秒）］ *PIM－SM Register Traffic Received*（*packets/sec*）［接收到的 PIM－SM 注册流量（报文数/秒）］ *PIM－SM Register Traffic Sent*（*bits/sec*）［发送的 PIM－SM 注册流量（比特/秒）］ *PIM－SM Register Traffic Sent*（*packets/sec*）［发送的 PIM－SM 注册流量（报文数/秒）］	这些统计量记录这个节点在其所有接口上接收或发送的 PIM－SM 注册消息总量。这些统计量以 bit/s 和报文数/s 记录

（续）

分　类	名　字	描　述
Node Statistics PIM - SM（节点统计量 PIM - SM）	*Network Convergence Activity*（网络收敛活动）	这个统计量记录在这个节点处时间收敛时段。收敛活动的时段被表示为1，而没有收敛活动的时段被表示为0。得到的图由多个1和0组成，对应于在这个节点处有收敛活动和没有收敛活动的时段
	Network Convergence Duration（网络收敛时长）	这个统计量记录针对一个特定组播组，这台路由器使其PIM路由表收敛的时间量。这个统计量被记录为从路由器展示收敛活动的迹象的时间直到没有收敛活动的时段时消逝的时间量

除了统计量外，会发现检查组播路由表也是有用的。具体而言，OPNET分别允许通过**Reports... IP Multicast Group - to - RP Table**（报告...IP组播组到RP表）和**Reports... PIM _ SM Routing Table**（报告...PIM _ SM路由表）属性输出组到RP映射和PIM - SM路由表。可通过**Results Browser**窗口中的**DES Run Tables**选项卡查看所输出的表。

10.3　IPv6

10.3.1　所支持IPv6功能特征概述

OPNET提供对IPv6（互联网协议版本6）的支持。但是，为了在一个仿真中使用IPv6模块，需要具有**IPv6 for R&D**模块许可证。OPNET支持各种IPv6功能特征，如在接口上指派IPv6地址，设置双栈节点配置（即各节点可被配置为支持IPv4和IPv6协议），指定IPv6静态路由表，创建IPv6隧道，部署某些IPv6路由协议，在一个IPv6网络内创建流量需求和标准应用源，产生ICMPv6 ping消息，部署QoS支持，通过移动IPv6连接移动节点，等等。但是，在IPv6网络内，OPNET不支持路径MTU发现，因此所有支持IPv6的接口都被配置为仅支持1500字节的MTU尺寸。在本节，仅描述基本的IPv6配置参数，原因是支持的许多IPv6功能特征是以类似于IPv4的方式进行配置的。欲了解IPv6功能特征的更完全描述，请通过**Protocols→IPv6→Model User Guide**选项参见IPv6用户指南。OPNET也包括称为IPv6的一个预配置项目，它提供了各种IPv6功能特征的绝佳展示。

10.3.2　IPv6编址

在OPNET中，支持层3功能的所有节点模型都支持IPv6。通过使用**Protocols→**

IPv6→Auto－Assign IPv6 Address 选项，可在网络中指定 IPv6 地址。具体而言，如果在这个操作之前，已经选择了节点和/或链路，那么 OPNET 将支持 IPv6（即将链路一本地 IPv6 地址设置为值 Default EUI－64），并将在还没有指派 IPv6 地址的选中节点和链路的所有相应接口上指派全局 IPv6 地址。如果在选择 **Auto－Assign IPv6 Address**（自动指派 IPv6 地址）选项之前，没有选择任何节点，那么 OPNET 将激活 IPv6，并将在网络中所有连接的接口上指定全局 IPv6 地址。通过使用 **Protocols→IPv6→Clear IPv6 Address**（协议→IPv6→清除 IPv6 地址）选项，可清除所指派的 IPv6 地址。这项操作清除所有指派的全局 IPv6 地址，并将所有链路本地 IPv6 地址设置为值 Not Active（不活跃的），这表明该接口没有激活 IPv6。

从 9.3 节，您可能记得，一个节点的接口上的一个 IPv4 地址是通过属性 **Address** 指定的。在端节点模型中，这个属性位于 **IP... IP Host Parameters... Interface Information**（IP... IP 主机参数... 接口信息）之下，而在核心节点模型中，**Address** 属性位于 **IP... IP Routing Parameters... Interface Information... < interface name >**（IP... IP 路由参数... 接口信息... <接口名>）之下。IPv6 地址是 128 比特长的，并通过称为 **Link－Local Address**（链路本地地址）和 **Global Address（es）... Address**（全局地址... 地址）的属性进行指定的，它们分别定义链路本地 IPv6 地址和全局 IPv6 地址。在端节点模型中，这些属性位于 **IP... IP Host Parameters... Interface Information... IPv6 Parameters**（IP... IP 主机参数... 接口信息... IPv6 参数）之下，而在核心节点模型中，它们位于 **IP... IPv6 Parameters... Interface Information... < interface name >**（IP... IPv6 参数... 接口信息... <接口名>）之下，如图 10.17 所示。为了指定一个链路本地 IPv6 地址，需要设置属性 **Link－Local Address**（链路本地地址），该属性支持如下值：

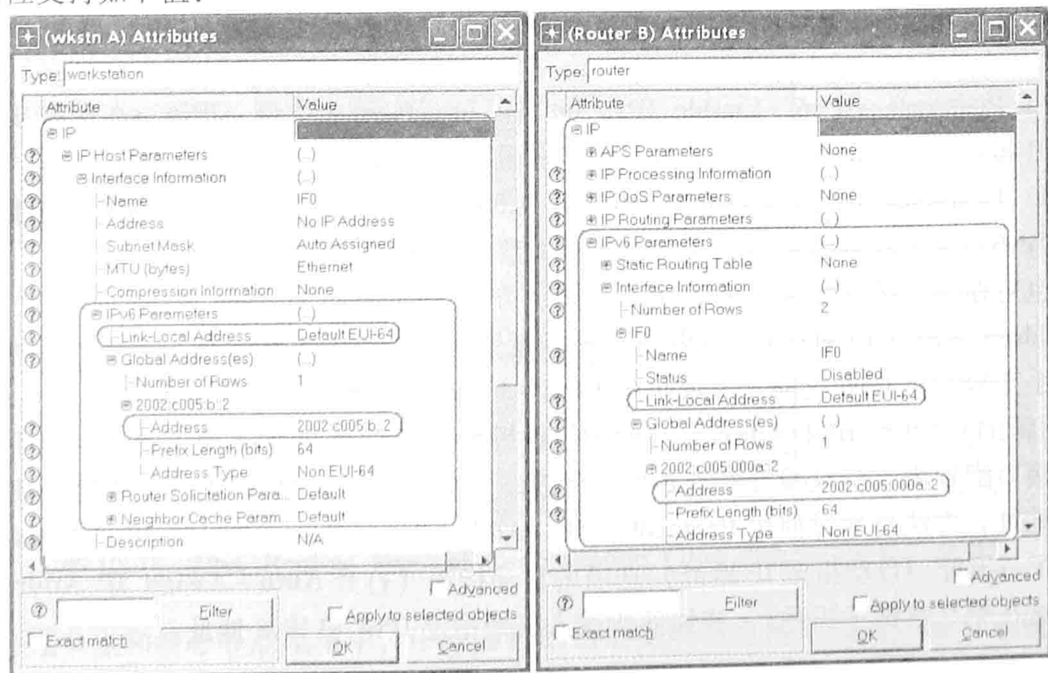


图 10.17　指定 IPv6 链路本地和全局地址

1）Default EUI－64（默认的 EUI－64）指明在这个接口上的链路本地 IPv6 地址将在仿真执行时自动地确定。这个值也指明该接口是激活 IPv6 的。

2）Not Active（未激活的）指明这个接口没有激活 IPv6。

3）Edit... 使您可手工指定 IPv6 值。但是，像在 RFC 2373 中声称的那样，一个手工指定的 IPv6 链路本地地址的第一个 8 字节必须对应于值 FE80::。注意所有链路本地地址的前缀长度均为 64 比特。

为在一个接口上指定一个全局 IPv6 地址，需要实施如下步骤：

1）展开子属性 **Global Address（es）**（全局地址）。

2）将属性 **Number of Rows** 的值设置为一个整数，该数对应于要在那个接口上配置的全局 IPv6 地址总数。注意每个接口可配置多个全局 IPv6 地址。

3）对于每个新建的行，需要将属性 **Address** 的值设置为一个全局 IPv6 地址。全局 IPv6 地址值是不能自动指派的，必须手工配置。但是，如果节点被配置为支持 DHCP，那么可将属性 **Address** 的值设置为 From DHCP（来自 DHCP）或 From DHCP Prefix Delegation（来自 DHCP 前缀委派），这将自动地得到 IPv6 地址。欲了解 IPv6 DHCP 配置指令，参见 *OPNET IPv6 Model User Guide*（OPNET IPv6 模型用户指南）。进而，如果 Address Type 属性的值设置为 EUI－64（扩展的唯一标识符格式），那么仅需要指定 IPv6 的前 64 比特。其他 64 比特是基于接口的 48 比特 MAC 地址值，按照 RFC 2373 建议的方法自动计算的。

典型情况下，每个层 3 节点模型可支持 IPv4 和 IPv6 协议。这种节点模型经常被称作支持双栈的。支持双栈的每个节点可配置为支持 IPv4、IPv6 或同时支持 IPv4 和 IPv6。使用如下 Protocols 菜单选项，可在节点接口上激活或禁止 IPv6 功能：

1）**Protocols→IPv6→Configure Interface Status...**（协议→IPv6→配置接口状态...）——打开一个窗口，使您在被选中节点和链路或所有节点上激活或禁止 IPv6 功能。

2）**Protocols→IPv6→Enable IPv6 on All Interfaces**（协议→IPv6→在所有接口上激活 IPv6）——在网络中的所有节点接口上激活 IPv6 功能。

3）**Protocols→IPv6→Disable IPv6 on All Interfaces**（协议→IPv6→在所有接口上禁止 IPv6）——在网络中的所有节点接口上禁止 IPv6 功能。

也可在一个接口上手工激活 IPv6 功能，方法是将 IPv4 **Address** 属性的值设置为 No IP Address（没有 IP 地址），并将 IPv6 地址设置为一个有效的 IPv6 值。类似地，可在接口上手工地禁止 IPv6 功能，方法是将 **Link－Local Address** 属性的值设置为 Not Active（不活跃的），并将 IPv4 **Address** 属性的值设置为一个有效的 IPv4 值。最后，也可将节点的接口配置为支持双栈设置，方法是在接口上指定 IPv4 和 IPv6 地址。可指派 IPv4 和 IPv6 地址，方法是通过使用 **Protocols→IP→Addressing→Auto－Assign IP Address**（协议→IP→编址→自动指派 IP 地址）选项的自动指派（打开 **Auto－Assign IP Address** 窗口，这在第 9 章做了描述），或如前所述人工地指定 IPv4 和 IPv6 地址。

10.3.3　为 IPv6 网络配置流量

OPNET 允许在没有任何额外配置步骤的条件下，在一个支持 IPv6 的网络中部署任何标准应用和流量需求。但是，为数据传递所用的 IP 版本取决于源和目的地节点的配置。具体而言，如果所有源和目的地节点是支持双栈的，那么位于 **Configure/Run DES** 窗口（见 4.3 节）的 **Inputs... Global Attribute**（输入 ... 全局属性）类中的属性 **IP... IP Version Preference**（IP... IP 版本首选）的值，确定用来传递应用数据的 IP 协议版本。默认情况下，**IP Version Preference** 的值设置为 IPv6。在源和目的地节点至少有一个不支持双栈的情况下，对所有节点而言共同的 IP 版本被用来传递应用数据。如果源和目的地节点不支持 IP 的一个共同版本，那么在这些节点之间就不能发送流量。

10.3.4　其他 IPv6 选项

OPNET 允许配置 IPv6 接口运行如下路由协议：RIPng、OSPFv3、IS - IS、BGP、AODV、DSR 和 OLSR。类似于 IPv4，可通过 **Protocols→IPv6→Configure IPv6 Routing Protocols...**（协议→IPv6→配置 IPv6 路由协议....）配置这些路由协议，该选项打开如图 10.18 所示的 **IPv6 Routing Protocol Configuration**（IPv6 路由协议配置）窗口。通过这个窗口，可指定希望部署哪些路由协议（即可能多于一个）和希望将这些路由协议部署到哪里（即所有接口，包括环回和隧道，仅有物理接口，或仅有被选链路上的物理接口）。欲了解有关人工配置 IPv6 路由协议的指令，请参见 *OPNET IPv6 Model User Guide*（OPNET IPv6 模型用户指南）。

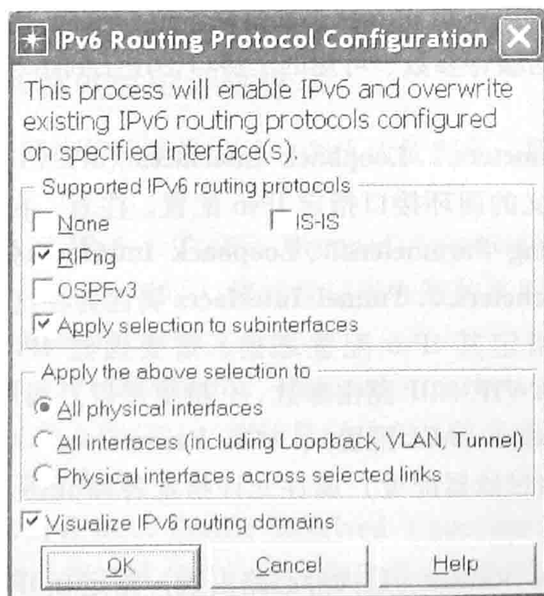


图 10.18　**IPv6 Routing Protocol Configuration**（IPv6 路由协议配置）窗口

在端节点和核心节点模型中，IPv6 参数的指定是稍稍有些不同的。端节点模型包含如下 IPv6 参数：

1）IP... IP Host Parameters... Interface Information... IPv6 Parameters（IP... IP主机参数... 接口信息... IPv6参数）属性使您可指定如下信息，如IPv6地址、路由器请求（solicitation）和邻居缓存参数。

2）**IP... IP Host Parameters... IPv6 Default Route**（IP... IP主机参数... IPv6默认路由）属性为这个端节点定义默认网关。指定的值必须表示一台直接连接的支持IPv6的路由器的一个有效IPv6地址。

3）**IP... IP Host Parameters... V6 Static Routing Table**（IP... IP主机参数... V6静态路由表）属性允许在端节点处定义一个IPv6静态路由表。这个属性主要用于这样的情形，即当端节点作为一台路由器时的情况（如在MANET环境中和/或当激活被动RIP配置时）。路由表表项是通过如下属性定义的，如**Destination Address**（目的地地址）、**Prefix Length**（前缀长度）、**Next Hop**（下一跳）和**Administrative Weight**（管理权重）。

核心节点模型支持比较广泛的IPv6功能特征，所有这些功能都被聚集在**IP... IPv6 Parameters**属性之下。具体而言，核心节点模型允许指定如下IPv6配置参数：

1）**IP... IPv6 Parameters... Static Routing Table**（IP... IPv6参数... 静态路由表）属性指定在这台路由器处部署的IPv6静态路由表。核心节点模型的每个路由表表项包含定义一条IPv6静态路由的附加属性，包括如下信息，诸如表项是否为永久的、VPN路由/转发（VRF）实例表的名字、组播反向路径转发（RPF）权重等。

2）**IP... IPv6 Parameters... Interface Information**（IP... IPv6参数... 接口信息）允许指定并配置路由器的支持IPv6的接口。一个支持IPv6的接口的配置，要求指定如下信息，如接口的名字、接口的IPv6状态、链路本地和全局IPv6地址、支持的IPv6路由协议、路由通告和邻居缓存参数、可用的子接口、用于到达和离开这个接口的报文的报文过滤器等。

3）**IP... IPv6 Parameters... Loopback Interfaces**（IP... IPv6参数... 环回接口）属性为在这个节点上定义的回环接口指定IPv6配置。注意，在指定其IPv6配置之前，需要通过**IP... IP Routing Parameters... Loopback Interfaces**属性定义回环接口。

4）**IP... IPv6 Parameters... Tunnel Interfaces**属性为在这个节点上定义的隧道接口指定IPv6配置。在指定其IPv6配置之前，需要通过**IP... IP Routing Parameters... Tunnel Interfaces**（IP... IP路由参数... 隧道接口）属性定义隧道接口。

5）**ACL Configuration**（ACL配置）、**ACL Parameters**（ACL参数）和**Prefix Filter Configuration**（前缀过滤器配置）属性允许指定各种IPv6访问控制和前缀过滤器列表。

6）**Maximum Static Routes**（最大静态路由数）指定在IPv6静态路由表中允许的最大表项数。

7）**General Prefixes**（通用前缀）属性指定用来定义较长的比较具体的IPv6前缀的各种短IPv6地址前缀。

8）**Local Policy**（本地策略）属性为这个接口指定一个路由表策略。但是，这个

属性目前不被支持，且对离散事件仿真没有影响。

除了 IPv6 节点属性和 **Protocols→IPv6**（协议→IPv6）选项以外，OPNET 也包含在整个场景中影响 IPv6 配置的全局 IPv6 属性。这些属性通过 **Configure/Run DES** 窗口中的 **Inputs... Global Attributes**（输入... 全局属性）类可进行访问。

1）**IP... IPv6 Configuration**（IP... IPv6 配置）属性允许在整个场景中激活或禁止 IPv6 配置。这个属性接受两个值：

① Consider（考虑）——强制仿真对配置的 IPv6 行为（即在仿真中激活 IPv6）进行建模。

② Ignore（忽略）——强制仿真忽略 IPv6 配置（即在仿真中禁止 IPv6）。

2）**IP... IPv6 Interface Address Export**（IP... IPv6 接口地址输出）属性允许将在被仿真网络中所有接口上指定的 IPv6 地址输出到一个文件。这个属性仅接受两个值：Enabled 和 Disabled，这分别对应于激活或禁止 IPv6 地址输出。被输出的 IPv6 地址被保存到一个逗号分隔纯文本文件，可采用任何文本编辑器或电子表格（spreadsheet）应用打开。被输出的文件名为 <project> - <scenario> - ipv6 _ addresses. gpf，其中 <project> 和 <scenario> 是这个项目和场景的名字。被输出的文件被保存在默认目录中。

3）**Simulation Efficiency... IPv6 ND Simulation Efficiency**（仿真效率... IPv6 ND 仿真效率）属性指定 DES 如何将 IPv6 地址转换为 MAC 地址。这个属性接受如下两个值：

① Enabled——强制仿真使用一个全局表，并可能使仿真执行得较快速。

② Disabled——强制仿真依赖于显式的邻居请求/通告消息来发现 IPv6 到 MAC 地址的映射。

10.3.5 IPv6 统计和其他性能评估选项

OPNET 为评估 IPv6 网络的性能提供了一个统计量集合。具体而言，仿真可被配置记录如下 IPv6 统计量：

1）**Global Statistics... IPv6... Traffic Dropped（packets/second）**［全局统计量... IPv6... 丢弃的流量（报文数/秒）］统计量记录在被仿真网络中所有节点中所有支持 IPv6 的接口上每秒钟丢弃的 IPv6 报文总数。

2）**Node Statistics... IPv6... Traffic Dropped（packets/second）**［节点统计量... IPv6... 丢弃的流量（报文数/秒）］统计量记录在当前节点中所有支持 IPv6 的接口上每秒钟丢弃的 IPv6 报文总数。

3）**Node Stattistics... IPv6... Traffic Received（packets/second）**［节点统计量... IPv6... 接收到的流量（报文数/秒）］统计量记录在当前节点中所有支持 IPv6 的接口上的网络处接收到的每秒 IPv6 报文总数。

4）**Node Statistics... IPv6... Traffic Sent（packets/second）**［节点统计量... IPv6... 发送的流量（报文数/秒）］统计量记录在当前节点中所有支持 IPv6 的接口上发送进入网络的每秒钟 IPv6 报文总数。注意这个统计量记录所有转发的报文，包括

后来由部署在节点接口上的 QoS 机制丢弃的那些报文。

OPNET 也为移动 IPv6 网络的性能评估提供全局统计量，以及记录由一个节点发送和接收的 IPv6 组播流量总量的节点统计量。OPNET 包含一个 IPv6 可视化选项，提供部署在网络内的 IPv6 路由协议的一个快照视图。可激活这项功能，方法是选择 View→Visualize **Protocol Configuration→IP Routing Protocols→IPv6 Routing Protocols**（视图→可视化协议配置→IP 路由协议→IPv6 路由协议）选项，将代表各种 IPv6 路由协议的各图标放置在这些协议要部署的链路上。

最后，通过实施如下动作，也可产生用户定义的 IPv6 报告：

1）从 **Project Editor** 中的下拉菜单中选择 **Scenarios→User – Defined Reports→Generate Report from Template...**（场景→用户定义的报告→从模板中产生报告...），这将打开 **Generate User – Defined Report** 窗口。

2）在如图 10.19 所示的 **Generate User – Defined Report** 窗口中，选择希望产生的 IPv6 报告。

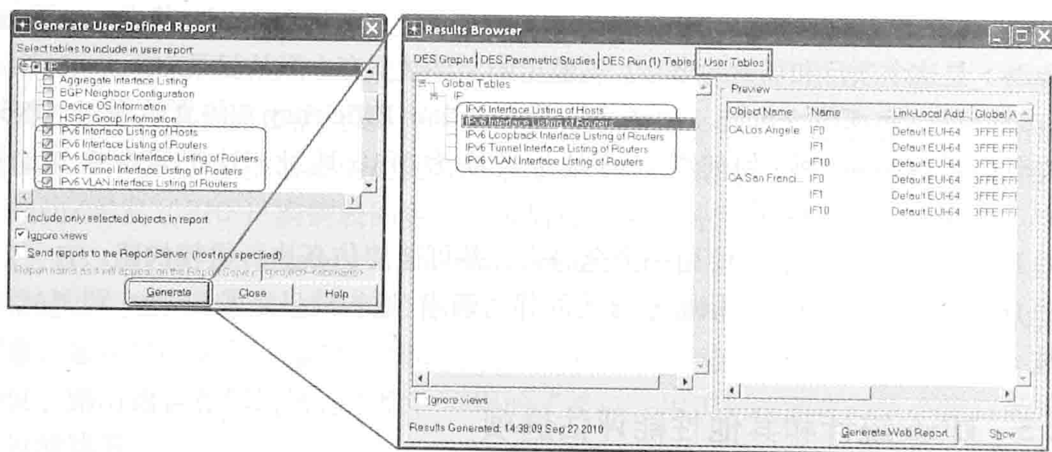


图 10.19 IPv6 用户定义的报告

3）一旦选择期望的报告，则单击 **Generate**（产生）按钮。

4）在执行仿真之后，通过 **Results Browser** 窗口（从下拉菜单中选择 **DES→Results→View Results**）中的 **User Tables**（用户表）选项卡，可访问所产生的报告。

10.4 服务质量

最初情况下，设计互联网主要用于网络节点之间的用户数据交付。但是，随着互联网应用的数量和类型增加，就如何通过网络交付数据的需求开始出现。互联网不仅要为数据交付负责，而且也要为性能准则或应用所经历的服务水平负责。典型情况下，这些服务水平是以取得的带宽和/或经历的延迟、抖动和/或丢失来定义的。进而，因为应用通常具有各种需求，所以互联网业需要在流量断续流及其需求之间做出区分，从而使每条流或流组接受到其期望的服务水平。一般而言，服务质量（QoS）可被定义为网络向

各种流量断续流保障某些服务水平的能力。OPNET 为提供 QoS 保障支持如下机制：

1）**Traffic Classifier**（流量分类器）——基于某些特征对流量断续流分类。

2）**Traffic Marker**（流量标记器）——将某些报文首部字段（如 ToS 字节）设置为后来可被用来识别报文的流量类的值。

3）**Traffic Policer**（流量监管器）——限制到达或离开一个接口的流量传输速率。流量监管规则会导致不符合的报文被丢弃、标记为不符合的或简单地不做处理。典型情况下，通过限制进入网络的流量总量，流量监管实现在网络边缘以防止拥塞。

4）**Congestion Avoidance mechanism**（拥塞避免机制）——当一个核心节点的缓冲占有状态达到某个阈值，概率性地丢弃到达的报文。丢弃到达报文的做法，强制相应的 TCP 源降低其拥塞窗口，这实际上减慢了其传输速率，并降低了拥塞概率。OPNET 支持随机早期检测（RED）和加权的 RED（WRED）机制。

5）**Traffic Scheduler**（流量调度器）——确定被缓冲在逻辑队列中的报文如何被调度离开队列。OPNET 支持如下调度机制：优先级排队（PQ）、低延迟队列（LLQ）、定制排队（CQ）、加权的轮转法（WRR）、加权的公平排队（WFQ）等。

当前，OPNET（即 IT Guru 和 Modeler 版本 16.0）不支持如下 QoS 机制，诸如基于流的 WFQ、基于流的 WRED、流量整形（即在流量符合速率限制规格之前，对流量的报文进行延迟处理）、速率限制访问控制列表、表映射和丢弃概要（profiles）。

在一个被仿真网络中部署 QoS 的过程，由两个主要步骤组成：①定义和配置 QoS 机制或概要；②在网络中的期望接口上部署定义好的 QoS 概要。

可在任意物理接口、子接口、聚合接口或 VLAN 接口上部署 QoS 概要。但是，不允许在环回接口或隧道接口上指定 QoS 概要。有两种类型的 QoS 概要：**local**（本地的）和 **global**（全局的）。通过在一个节点处改变属性值，可定义本地 QoS 概要。一个本地 QoS 概要仅可由这个概要被定义的节点访问，即在网络中的各节点仅可访问其自己的本地 QoS 概要。另外，全局 QoS 概要是场景范围的，即在该场景内的所有节点均可访问全局 QoS 概要。

在本节，讨论配置和部署 QoS 概要的步骤。具体而言，本节后面部分如下组织。首先，简短地介绍各种 QoS 机制，并讨论每种机制如何可被定义为一个全局 QoS 概要。接下来，将讨论配置本地 QoS 概要的步骤，接着是在网络中部署所定义 QoS 概要的指令。以评估 QoS 性能的可用统计量综述结束本节。在对所支持 QoS 功能特征的建模方面而言，看来 OPNET 密切遵循 Cisco 标准。就各种 QoS 功能特征和某些 QoS 配置属性的含义方面，会发现参考 Cisco 文档是有用的。

10.4.1 指定全局 QoS 概要

为指定全局 QoS 概要，需要将称为 *QoS Attribute Config*（QoS 属性配置）的一个工具对象添加到项目工作空间。可在 **Object Palette Tree** 的 *internet _ toolbox*、*QoS* 和 *utilities* 对象调色板中找到 *QoS Attribute Config* 对象。通过 *QoS Attribute Config* 对象可定义所有全局 QoS 概要，每个场景仅需要一个这样的对象。如图 10.20 所示，*QoS Attribute*

Config 对象由如下复合属性组成：

1）**CAR Profiles**（CAR 概要）为流量监管定义全局承诺的访问速率（CAR）概要。

2）**Custom Queuing Profiles**（定制排队概要）、**FIFO Profiles**（FIFO 概要）、**MWRR/MDRR/DWRR Profiles**（MWRR/MDRR/DWRR 概要）、**Priority Queuing Profiles**（优先级排队概要）、**WFQ Profiles**（WFQ 概要）为各种报文调度机制指定全局 QoS 概要。

3）**RSVP Flow Specification**（RSVP 流规格）和 **RSVP Profiles**（RSVP 概要）指定全局 RSVP 配置。在本书中不讨论集成服务和 RSVP。

下面讨论定义全局监管器和调度器概要的每个复合属性。

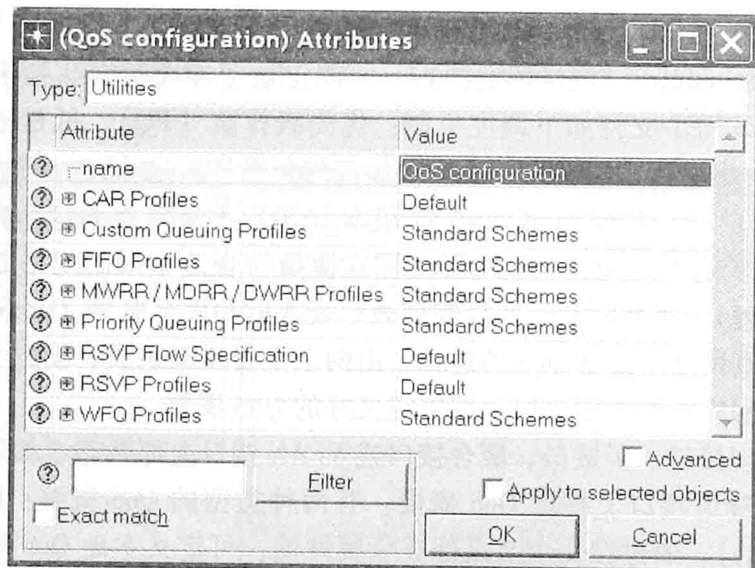


图 10.20 *QoS Attribute Config* 对象

1. 承诺的访问速率概要

可使用如图 10.21 所示的 **CAR Profiles** 属性，为流量监管指定全局 CAR 概要。默认情况下，OPNET 提供两个预配置 CAR 概要：限制所有流量传输速率的一个概要和仅抑制 HTTP 流量的另一个概要。另外，**CAR Profiles** 属性包含一个空 CAR 概要，可被用来进行配置。如果期望的话，可通过改变属性 **Number of Rows** 的值，指定其他的 CAR 概要。每个 CAR 概要定义由如下属性组成：

1）**Profile Name**（概要名）提供这个概要的一个描述性名字。这个属性的值识别这个 QoS 概要，当将概要部署在网络中时使用该名字。

2）**Details**（细节）包含当前 CAR 概要的一个实际配置。

具体而言，在属性 **Details** 下，可指定多个服务类（COS）概要（即通过属性 **Number of Row** 控制 COS 概要数量）。每个 COS 概要由一个流量分离器和一个流量监管器组成，通过如下属性进行配置：

1）**COS Definition**（COS 定义）配置流量分类器，方法是为确定到达报文是否属于当前 COS 而指定规则。COS 可依据 TOS 字节（也被称作 DSCP 值或优先级值）、应用

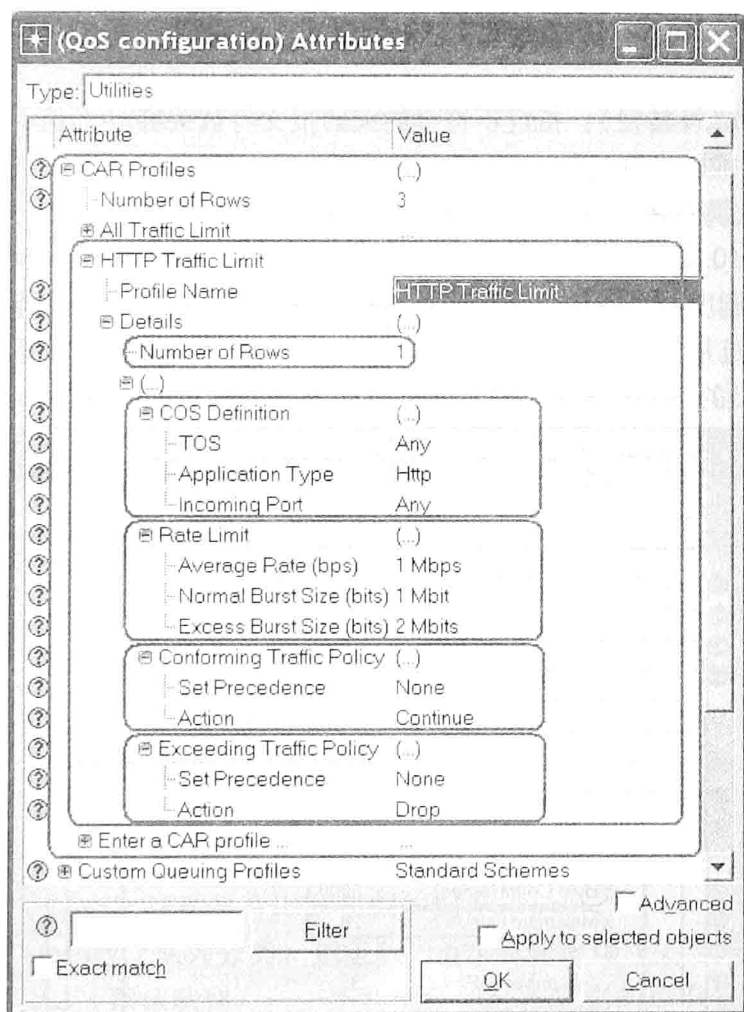


图 10.21　一个 **CAR Profiles** 属性

类型和/或到达端口号对要分类的到达报文进行配置。

2）**Rate Limit**（速率限制）为当前 COS 指定速率限制。通过诸如平均速率以及正常和过线（excess）突发尺寸等准则，定义速率限制。以低于平均速率的一个速率或低于正常突发尺寸到达的流量，被认为是符合约束的。以高于正常突发尺寸但小于过线突发尺寸的速率而到达流量突发，被看作是以随着流量突发尺寸增加而增加的一个概率不符合约束的。以大于过线突发尺寸的一个速率到达的所有流量被看作以概率 1 不符合约束的。

3）**Conforming Traffic Policy**（符合约束的流量策略）为处理符合约束的流量而配置规则。具体而言，策略由两个步骤组成：

① **Marking**（标记）：需要指定符合约束的报文（如果存在的话）如何改变其优先级值（即 ToS 字节值）。

② **Forwarding**（转发）：需要指定符合约束的报文将被立刻传输（即值 Transmit）还是被转发到当前 CAR 概要内的下一个 COS 定义（即值 Continue）。

4）**Exceeding Traffic Policy**（超过流量策略）为处理不符合约束的流量而配置规则。同样，越界的（exceeding）流量策略规则由两个步骤组成：

① **Marking**（标记）：指定不符合约束的报文（如果存在的话）将如何改变它们的优先级值。

② **Forwarding**（转发）：指定不符合约束的报文将被丢弃（即值 Drop）、离开传输（即值 Transmit）或转发到当前 CAR 概要内的下一个 COS 定义（即值 Continue）。

2. 定制排队调度器

可通过如图 10.22 所示的属性 **Custom Queuing Profiles**（定制排队概要）来配置 CQ 调度器。CQ 调度器以一种轮转方式处理逻辑队列。在一个队列的轮次中，CQ 调度前处理该队列的流量份额。在 OPNET 中，一个逻辑队列的流量份额是通过 **byte count**（字节计数）定义的，它指定了在队列的轮次中从队列传输的字节数。

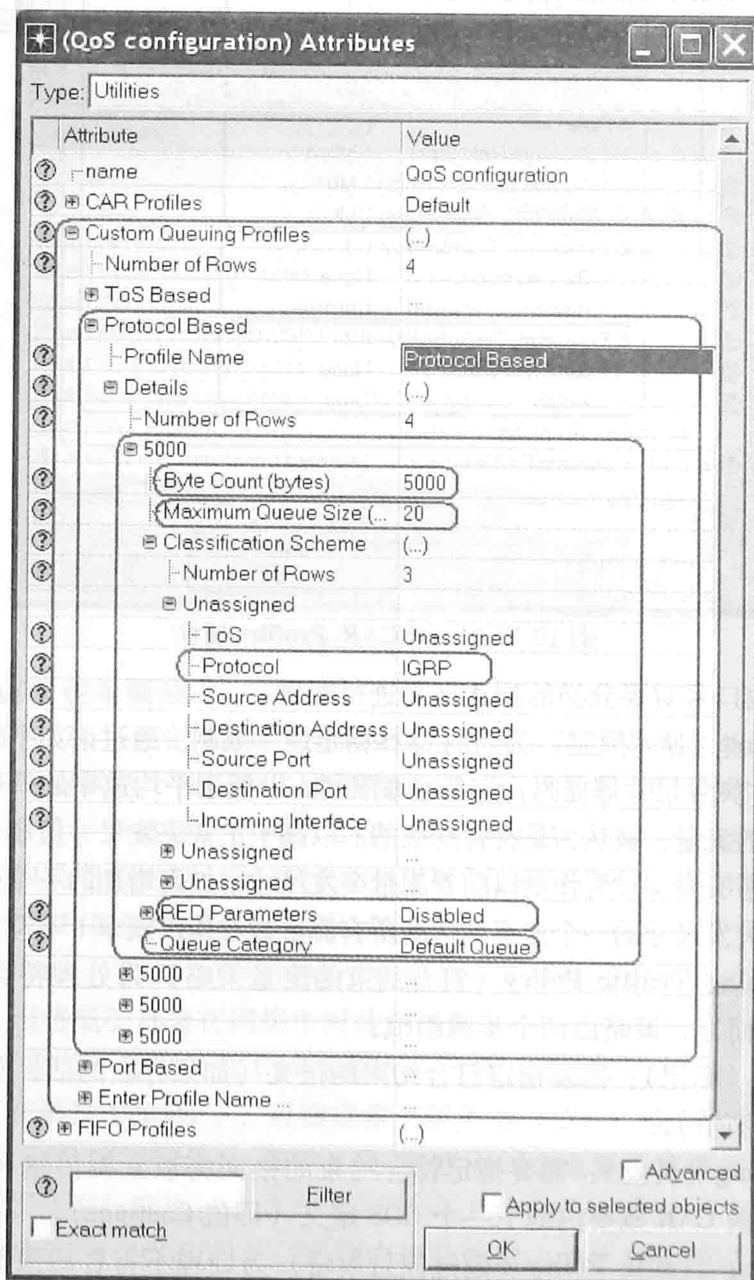


图 10.22　**Custom Queuing Profiles**（定制排队概要）属性

默认情况下，OPNET 提供了三个预设的 CQ 概要配置和一个准备被配置的空 CQ 概要。每个 CQ 概要定义由如下内容组成：

1）**Profile Name**（概要名）是一个基本属性，指定一个描述性的概要名，用来在场景内识别这个概要。

2）**Details**（细节）是一个复合属性，包含这个 CQ 概要的实际定义。

通过属性 **Details**，可配置 CQ 调度器内的每个个体逻辑队列。通过 **Number of Rows** 属性控制 CQ 调度前中逻辑队列的总数，且每个逻辑队列是通过如下属性进行配置的：

1）**Byte Count**（**bytes**）[字节计数（字节）] 为在该队列的轮次中，由该队列要传输的流量总量。当服务一个队列时，直到发送的字节数大于属性 **Byte Count**（**bytes**）的值或直到队列变空之前，进行报文传输。

2）**Maximum Queue Size**（**pkts**）[最大队列尺寸（报文）] 指定在拥塞过程中在这个队列中被允许的报文总数（即在拥塞过程中，如果在这个逻辑队列中的报文数超过这个属性的值，那么到达这个队列的所有报文都将被丢弃）。

3）**Classification Scheme**（分类方案）为识别要放入到这个逻辑队列中的报文，指定分类规则。可基于报文的 ToS 值、协议类型、源 IP 地址、目的地 IP 地址、源端口号、目的地端口号和/或到达接口识别报文。这个属性允许指定多个分类规则。通过属性 **Number of Rows** 控制规则总数，每个新建的行代表一条独立的分类规则。

4）**RED Parameters**（RED 参数）允许配置在这个队列上部署的 RED 或 WRED 拥塞避免机制。后面会讨论 RED 和 WRED 机制的配置。

5）**Queue Category**（队列分类）指定队列类型。这个属性接受如下值：

① Default Queue（默认队列）——不匹配任何分类规则的所有报文被放置到默认队列。对于每个物理队列，仅有一个默认队列（即在调度器内不能配置多个队列作为默认队列）。

② Low Latency Queue（低延迟队列）——在所有其他流量之前服务在 LLQ 中的报文，仅当 LLQ 为空时，其他队列在依据 CQ 调度机制被处理。LLQ 提供对延迟敏感流量（如话音或视频）的优先处理。对于每个物理队列，仅有一个 LLQ（即在调度器内不能配置多个队列作为 LLQ）。

③ Default and Low Latency Queue（默认的和低延迟队列）——队列是默认的和低延迟的。

④ None（无）——队列既不是默认的也不是低延迟的。

例外情况是 **Byte Count**（**bytes**），配置 CQ 调度器中逻辑队列的所有属性也用于配置 PQ 中的逻辑队列 [即 **Priority Queuing Profiles**（优先级排队概要）]、WRR [即 **MWRR/MDRR/DWRR Profiles**（MWRR/MDRR/DWRR 概要）] 和 WFQ [即 **WFQ Profiles**（WFQ 概要）]。这就是没有再次提供这些属性的描述的原因。

3. RED 和 WRED 配置

如上面所述，对任何全局调度机制概要的逻辑队列配置属性，包括属性 **RED Pa-**

rameters（RED参数），允许在一个相应的逻辑队列上配置RED/WRED拥塞避免机制。RED机制的思想是非常简单的：当队列占有率达到某个水平，通过标记或丢弃到达的报文，将可能的拥塞通知源。具体而言，RED维持平均队列占有率的一个估计，我们称之为 *avg*。当平均队列占有率达到最小阈值 $\min_{th}$，RED以称之为报文—标记概率的某个概率开始标记报文。当平均队列占有率超过最大阈值 $\max_{th}$，那么RED标记所有到达报文。典型情况下，被标记的报文将其ECN字段（即在IP首部中ToS字节的最右边两个比特）设置为遇到拥塞（CE）值，这就将网络中的拥塞通知了网络边缘（设备）。其他的RED实现丢弃报文，而不是标记报文，这实际上也将网络中的拥塞通知给了TCP源。RED使用加权移动平均（EWMA）算法估计平均队列占有率。报文标记概率 p_b 是以式（10.1）计算的，其中 $p_{\max}$ 是RED配置参数，它指定了当队列占有率达到 $\max_{th}$ 时报文—标记概率的最大值。

$$p_b = p_{\max}\frac{\mathrm{avg} - \min_{th}}{\max_{th} - \min_{th}} \tag{10.1}$$

WRED是RED的一个变种，为具有不同ToS值的报文，它使用配置参数的一个不同集合（即 $\min_{th}$、$\max_{th}$ 和 $p_{\max}$）。在OPNET中，针对全局调度器概要的WRED配置，仅允许改变 $p_{\max}$ 的值，而 $\min_{th}$ 和 $\max_{th}$ 的值对所有报文是相同的。如图10.23所示，RED/WRED机制是通过 **RED Parameters** 的如下属性进行配置的：

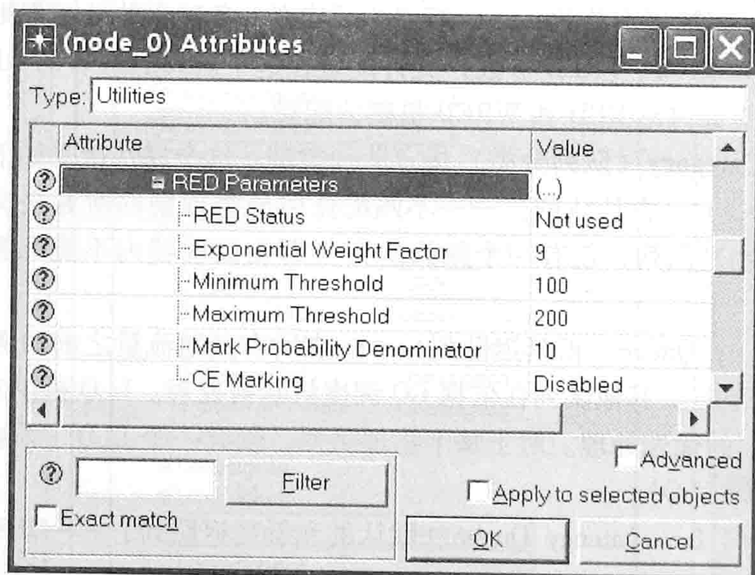


图10.23 属性 **RED Parameters**

1）**RED Status**（RED状态）——指定激活哪种机制：RED、WRED或两种都不激活。

2）**Exponential Weight Factor**（指数加权系数）——用于估计平均队列占有率的一个值。

3）**Maximum Threshold**（最大阈值）——最大队列占有率阈值。

4）**Minimum Threshold**（最小阈值）——最小队列占有率阈值。

5）**Mark Probability Denominator**（标记概率标准）——一个最大报文标记概率的标准。实际的最大标记概率 p_{max} 的计算为 1/（**Mark Probability Denominator** 属性的值）。当配置 WRED 时，可输入以逗号分隔的 **Mark Probability Denominator** 的多个值。

6）**CE Marking**（CE 标记）——当这个属性设置为 Enabled 时，确定被标记的报文，将它们在 IP 首部中的 ECN 字段设置为 CE 值。当这个属性被设置为 Disabled 时，就丢弃这种报文。

注意，OPNET 允许配置本地 WRED 概要，它支持全部 WRED 配置（即可指定 min_{th}、max_{th}和 p_{max}参数的多个集合）。

4. FIFO 概要

OPNET 允许指定先入先出（FIFO）调度概要。FIFO 概要定义的结构与所有其他调度器的方式相同，并由两个属性组成：**Profile Name**（概要名）和 **Details**（细节）。因为 FIFO 队列没有被分为逻辑子队列，属性 Details 通过属性 **Maximum Queue Size（pk-ts）**［最大队列尺寸（报文）］和 **RED Parameters**（RED 参数）配置 FIFO 调度器。这些属性与上面第 2 部分描述的 CQ 调度器的相应属性具有相同含义。

5. MWRR/MDRR/DWRR 概要

OPNET 也支持 WRR 调度器的三个变种：修正的加权轮转法（MWRR）、修正的赤字轮转法（MDRR）和赤字加权轮转法（DWRR）。所有这些机制依赖于一个赤字计数器，以字节为单位指定在每个轮转处理周期过程中可被服务的数据量。只要队列的赤字计数器大于零，则 MWRR 和 MDRR 就服务来自一个非空队列的报文。注意在后续轮转过程中，为了补偿前些轮转过程中处理的超额流量，队列可服务较少报文。另外，只要赤字计数器值大于要被处理的报文尺寸，DWRR 就服务报文，这意味着在后续轮转过程中，队列可服务较多数据，以补偿在前些轮转中的服务赤字。在轮转处理的每个轮次之后，赤字计数器增加正比于队列权重的一个值。MWRR 和 MDRR 以字节为单位维持赤字计数器，而 DWRR 以 53 字节（是一个 ATM 信元的尺寸）为单位维持赤字计数器。

为了配置如图 10.24 所示的一个全局 MWRR/MDRR/DWRR 概要，需要实施如下步骤［注意，属性 **MWRR/MDRR/DWRR Profiles**（MWRR/MDRR/DWRR 概要）和 **Custom Queuing Profile**（定制排队概要）的结构相同］：

1）指定希望创建的概要数。默认情况下，属性 **MWRR/MDRR/DWRR Profiles** 包含四个预定义的概要和一个以备配置的空概要。

2）对于每个概要，指定名字［即属性 Profile Name（概要名）］和队列配置的细节［即属性 **Queues Configuration**（队列配置）］。

3）队列配置要求指定 WRR 概要中逻辑队列的数量（即属性 **Number of Rows**），并独立地配置每个逻辑队列。

4）针对 MWRR/MDRR/DWRR 概要配置逻辑队列的所有属性与 CQ 概要中的相同。唯一的例外是属性 **Weight**（权重），它指定在当前逻辑队列执行轮次时为服务该队列而分配的带宽量。在 MWRR 的情形中，**Weight** 代表链路带宽的百分比，而对于 MDRR/

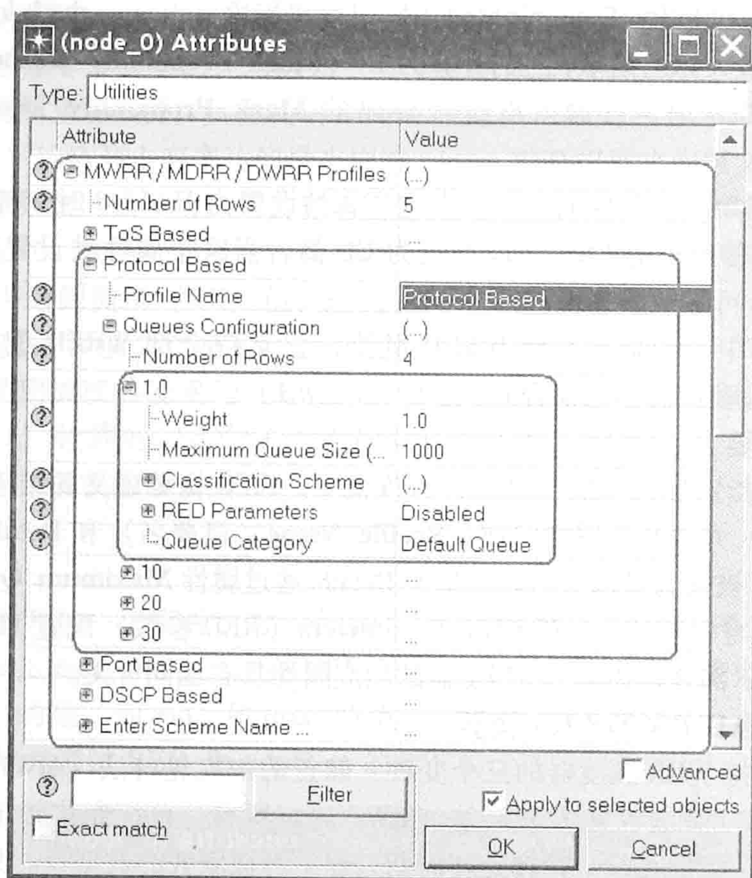


图 10.24　属性 **MWRR/MDRR/DWRR Profiles**

DWRR，属性 Weight 的值是相应接口上 MTU 的整数倍。

注意，为一个给定的概要定义（即 MWRR、MDRR 或 DWRR），没有指定轮转机制类型的属性。事实上，MWRR、MDRR 和 DWRR 调度器的全局概要是通过属性的相同集合进行配置的。但是，当在网络中部署 QoS 概要时，OPNET 要求首先指定 QoS 概要的类型［即 Custom Queuing（定制排队）、MWRR、FIFO、DWRR 等］，之后才是其名字。在 10.4.3 节讨论这些配置步骤。

6. 优先级排队概要

通过修改 **Priority Queuing Profiles**（优先级排队概要）属性，可配置 PQ 调度机制。PQ 机制以队列的优先级顺序服务各队列：仅当所有较高优先级队列为空时，才服务低优先级队列。OPNET 以配置全局 CQ 和 MWRR/MDRR/DWRR 概要属性的相同方式，组织 **Priority Queuing Profiles** 属性的结构。在 CQ 和 WRR 概要中个体队列的配置分别依赖于属性 **Byte Count**（**bytes**）［字节计数（字节）］和 **Weight**（权重），它们确定了专用于每个逻辑队列的资源份额。类似地，PQ 机制包含属性 **Priority Label**（优先级标签），指定一个逻辑队列的优先级。这个属性是 PQ、CQ 和 WRR 调度器之间的唯一配置差异。

7. WFQ概要

WFQ依据与每个逻辑队列相关联的权重值，在逻辑队列间提供公平的带宽分配。同样，以类似于全局CQ、WRR和PQ概要的方式，OPNET组织全局WFQ概要的结构。与以前一样，每个WFQ概要配置由指定概要名和逻辑队列配置的属性组成。另外，每个WFQ概要包含属性**Buffer Capacity**（缓冲容量），它以报文数为单位指定接口上的缓冲尺寸，其中接口指将在其上部署相应WFQ概要的接口。

在全局WFQ调度器概要中配置逻辑队列的属性，与如图10.24所示全局MWRR/MDRR/DWRR概要的那些属性相同。唯一差异是属性**Maximum Queue Size**（**pkts**）[最大队列尺寸（报文数）]和**Weight**（权重）的含义。在WFQ调度器中，属性**Maximum Queue Size**（**pkts**）确定最大报文数，这是指当在物理队列中的报文数达到属性**Buffer Capacity**的值时可在逻辑队列中积累的报文数量。具体而言，只要总的物理缓冲占有率没有超过**Buffer Capacity**的值，一个个体逻辑队列就可依据需要积累多条报文。一旦缓冲占有率达到**Buffer Capacity**，所有到达报文都被丢弃，直到逻辑队列占有率降低到**Maximum Queue Size**（**pkts**）或缓冲占有率降低到**Buffer Capacity**值之下时才不丢弃报文。注意，即使当超过**Buffer Capacity**时，OPNET也不丢弃OSPF和IGRP报文。在全局WFQ概要中，属性Weight为相应队列指定所分配带宽的份额，即较大的权重值对应于分配带宽的较大份额。

10.4.2 指定本地QoS概要

通过如图10.25所示的IP...IP QoS Parameters属性，可配置本地QoS概要，并部署所定义的QoS概要（即本地的和全局的）。端节点模型和核心节点模型包含配置QoS概要的相同属性集合。但是，即使节点模型可能包含配置那些不被支持的功能特征，这些功能特征也还没有得到支持。具体而言，通过如下属性定义的QoS功能特征还没有进行实现：**Traffic Shaping Profiles**（流量整形概要）、**Dropping Profiles**（丢弃概要）、**Rate - Limit Access Control Lists**（速率受限的访问控制列表）和**Table Maps**（表映射）。在仿真过程中忽略这些属性。在本节，提供配置各种本地QoS概要所需步骤的简短概述。

1. 流量类和流量策略

分别通过如图10.26和图10.27所示的**Traffic Classes**（流量类）和**Traffic Policies**（流量策略），可定义流量类并指定适用于已定义类的QoS策略。为定义新的流量类，需要实施如下两个步骤：

1）通过设置位于**Traffic Classes**属性下的**Number of Rows**的值，指定流量类的数量。

2）配置每个新建的流量类。

在第二步骤中，通过设置如下属性配置一个流量类：

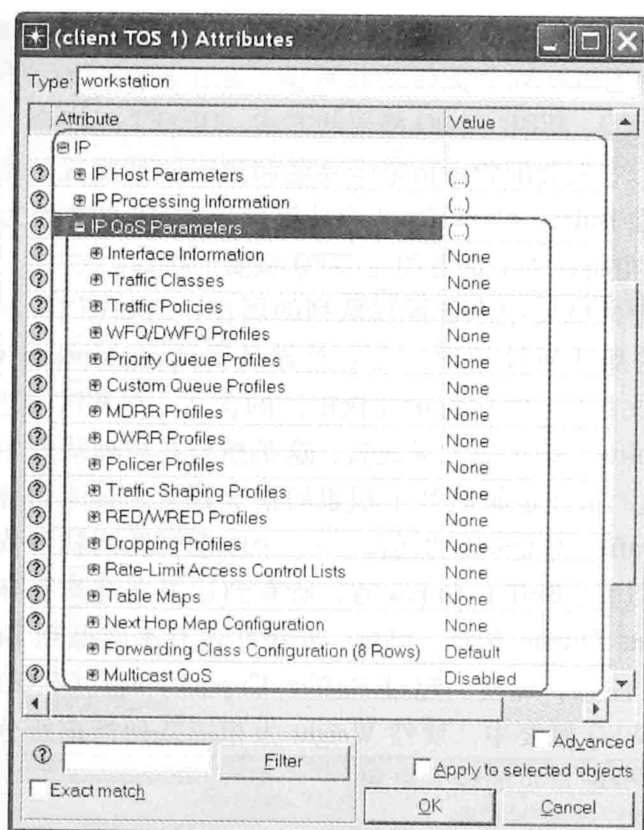


图 10.25　属性 **IP　QoS　Parameters**

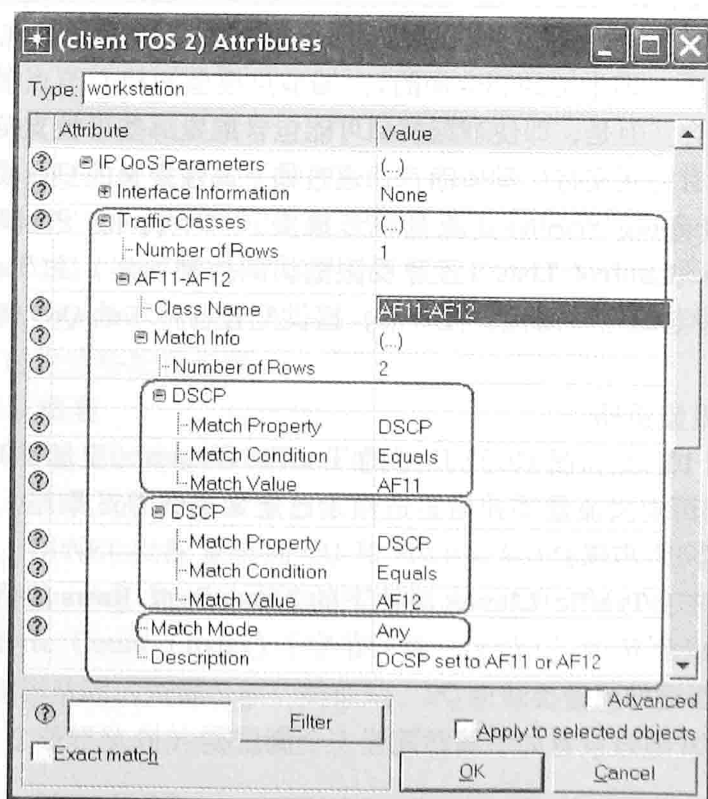


图 10.26　属性 **Traffic　Classes**

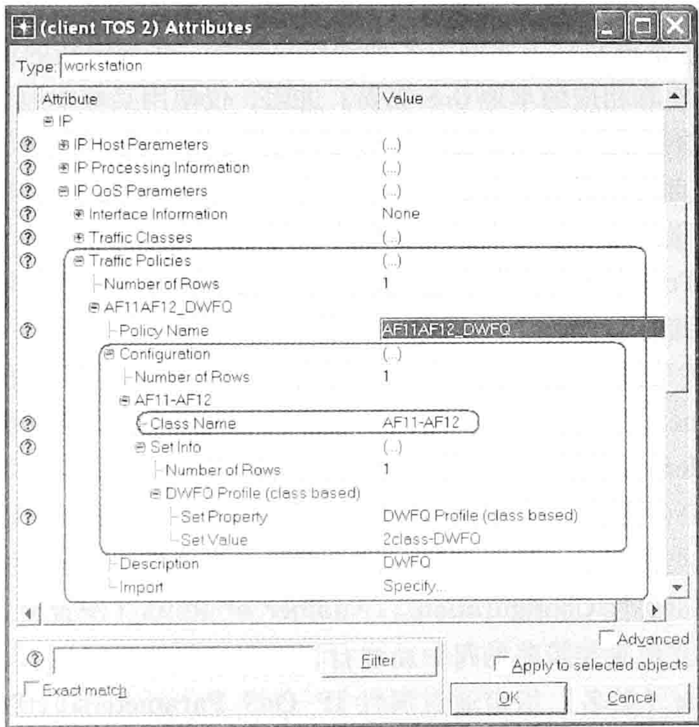


图 10.27　属性 **Traffic Policies**

1）**Class Name**（类名）是类的名字。这是在这个节点内用来识别当前流量类的名字。

2）**Match Info**（匹配信息）为一个复合属性，为当前流量类指定分类规则。如果需要可定义多条规则。对于每条规则，需要设置如下属性：

① **Match Property**（匹配性质）将识别当前类的流量性质［即 DSCP、Incoming Interface（到达接口）、Packet Size（报文尺寸）等］。

② **Match Condition**（匹配条件）用来比较 **Match Property** 和 **Match Value** 的一个条件，如 Equals（等于）、Greater Than（大于）、Less Than（小于）、Not Equals（不等于）。

③ **Match Value**（匹配值）用来识别这个流量类的匹配性质的值。

3）**Match Mode**（匹配模式）是为了被分类为这个流量类的组成部分，指定到达的报文必须满足所有分类规则（值 All）或至少其中之一（值 Any）。可以这样认为，将这个属性值设置为 All，就相当于创建单条分类条件（其中所有个体规则由逻辑 AND 连接），而将这个属性设置为值 Any，就相当于将个体分类规则以逻辑 OR 连接在一起。必须谨慎从事，以避免产生导致空流量类的分类规则。例如，考虑这样一种情况，其中 **Match Mode** 被设置为 All，Match Info 属性配置如下两条规则：DSP 值等于 AF11 的报文和 DSCP 值等于 AF12 的报文。因为一条报文的 DSCP 值不能同时等于 AF11 和 AF12，所以将没有报文匹配到这个流量类。

4）**Description**（描述）提供了这个流量类的简短描述，且不影响仿真执行。

流量策略指定对当前节点中各种流量类的处理。流量处理可包括要在本地 QoS 概要（应用到流量类的各条报文）上实施的各种动作。注意，在指定新的流量策略之前，需要定义必要的流量类和相应的本地 QoS 概要。进而，仅使用某些本地 QoS 概要指定流量策略。具体而言，不能使用任意全局 QoS 概要以及本地 PQ 和 CQ QoS 概要来指定流量策略。这些概要被直接部署在节点的接口上，将在 10.4.3 节描述该项内容。

为创建新的流量策略，需要实施如下动作：

1）展开属性 **Traffic Policies**（流量策略），并设置属性 **Number of Rows** 的值，该值对应于要定义的流量策略的总数。

2）为每个新建的策略行，指定如下属性值：

① **Policy Name**（策略名）——用来在当前节点内识别这个流量策略的名字。

② **Configuration**（配置）——实际的策略配置。

③ **Description**（描述）——这个流量策略的文本描述。

可为多个流量类指定单一流量策略（即设置属性 **Configuration** 的值）。由策略覆盖的流量类的属性是由属性 **Configuration... Number of Rows**（配置... 行数）加以控制的。为代表一个独立流量类策略的每个新建行，需要配置如下属性：

1）**Class Name**（类名）指定通过属性 **IP QoS Parameters... Traffic Classes**（IP QoS 参数... 流量类）定义的流量类的名字。

2）**Set Info**（集合信息）指定要对这个流量类实施的策略。

① **Number of Rows**（行数）控制为这个流量类定义的策略数量。

② 针对每行，需要指定如下内容：

- **Set Property**（集合性质）指定要应用到这个类之报文的性质。目前，OPNET 仅支持如下集合性质：WFQ Profile（WFQ 概要）、DWFQ Profile（DWFQ 概要）、DWRR Profile（DWRR 概要）、MWRR Profile（MWRR 概要）、Policer Profile（治理器概要）、RED/WRED Profile（RED/WRED 概要）、DSCP 和 Precedence（优先级）。
- **Set Value**（集合值）指定要应用到这个流量类的集合性质的值或名字。

2. WFQ/DWFQ 概要

OPNET 也支持本地 WFQ 概要。具体而言，可定义分布式 WFQ（DWFQ）和基于类的 WFQ（CBWFQ）调度机制概要。但是，基于流的 WFQ 概要目前在 OPNET 中是没有支持的。即使在节点模型中的 QoS 配置属性（即 **PIP QoS Parameters**）包括复合子属性 **Flow - based WFQ/DWFQ Profiles**（基于流的 WFQ/DWFQ 概要），在仿真过程中也忽略其配置值。

DWFQ 或 VIP - DWFQ 是运行在多功能接口处理器（VIP）之上 WFQ 机制的一个变种。VIP - DWFQ 得到某些 Cisco 路由器的支持，在本书中不做讨论。CBWFQ 是在流量类上工作的 WFQ 的一个变种。每个本地 CBWFQ 概要代表 CBWFQ 调度器内一个流量类的单个 FIFO 队列的配置。实际的 CBWFQ 调度器是通过流量策略定义的［即通过属性 **Traffic Policies**（流量策略）配置的］，它指定流量类［通过属性 **Traffic Classes**（流量类）］和本地 CBWFQ 概要之间的映射。为创建 CBWFQ 概要，需要实施如下步骤：

1）展开属性 IP QoS Parameters... Class - based WFQ Profiles（IP QoS 参数... 基于类的 WFQ 概要）。

2）设置属性 Number of Rows 的值，它代表 CBWFQ 概要的总数。

3）配置每个新建的行（对应于 CBWFQ 调度器内一个流量类的单个 FIFO 队列），方法是指定如下属性值：

① **Name**（名字）为 CBWFQ 概要的名字。

② **Bandwidth Type**（带宽类型）指定在拥塞过程中如何表示一个流量类的最小带宽保障：Relative（相对的）——链路容量的百分比，Absolute（决定的）——以比特每秒为单位的实际值，或 Remaining Percentage（剩余的百分比）——在所有其他队列已经得到其带宽份额之后，剩下的带宽百分比。

③ **Bandwidth Value**（带宽值）指定为一个流量类所保障的最小带宽的实际量。取决于 **Bandwidth Type** 值，这个属性接受以比特每秒为单位的绝对带宽量，或链路容量的百分比。

④ **Priority**（优先级）属性激活或禁止在这个队列上的优先级排队（即 LLQ）。如果激活优先级排队，那么为了防止 CBWFQ 调度器内所有其他队列出现饥饿状态，属性 **Bandwidth Value** 的值作为这个队列的一个速率限制。

⑤ **Queue Limit（packets）**［队列限制（报文数）］以报文数为单位指定最大队列容量。

⑥ **Burst Size（bytes）**［突发尺寸（字节）］和 **Default Class Dynamic Queues**（默认类动态队列）属性目前还不支持。

3. 优先级队列概要和定制队列概要

为了指定一个本地 PQ 概要，需要展开 **Priority Queue Profiles**（优先级队列概要）属性，设置 **Number of Rows** 属性的值（代表概要的总数量），之后通过设置如下属性配置每个个体 PQ 概要：

1）**List Number**（列表号）指定当前 PQ 概要的名字。

2）**Configuration**（配置）指定报文分类规则，它们确定一条报文要被放入的队列以及个体队列的配置细节。通过设置 **Number of Rows** 属性的值，可在单个 PQ 概要内指定多达 4 个队列。每个逻辑队列由如下配置属性组成：

① **Queue Priority**（队列优先级）指定当前队列的优先级。这个属性仅接收值 Low（低）、Normal（正常）、Medium（中等）和 High（高）。

② **Match Info**（匹配信息）为识别要在这个队列中存储的报文而指定规则。

③ **Queue Limit（packets）**［队列限制（报文数）］以报文数为单位指定这个队列的最大容量。

3）**Default Queue**（默认队列）指定概要内的哪个队列（即 Low、Normal、Medium 或 High）将作为一个默认队列。默认队列接受不匹配 PQ 概要内任何分类规则的所有报文。

本地 PQ 和 CQ 概要的定义具有一个非常类似的结构。为指定一个本地 CQ 概要，也

需要展开 **Custom Queue Profiles**（定制队列概要）属性，设置 **Number of Rows** 属性的值（代表概要的总数），之后通过指定如下属性值而配置每个个体 CQ 概要：

1）**List Number**（列表号）指定 CQ 概要的名字。

2）**Configuration**（配置）指定报文分类规则，该规则确定将一条报文放置的队列和个体队列的配置细节。通过设置 Number of Rows 属性的值，可在 CQ 概要内指定多达 16 个队列。每个队列由如下配置属性组成：

① **Queue Number**（队列号）指定当前队列的号。这个属性接受 1 和 16 之间的任意整数。

② **Match Info**（匹配信息）为识别要在这个队列中存储的报文指定规则。

③ **Number of Bytes Allowed**（bytes）[允许的字节数（字节）] 以字节为单位指定每个轮次过程中由这个队列服务的流量总量。

④ **Queue Limit**（packets）[队列限制（报文数）] 以报文数为单位指定这个队列的最大容量。

3）**Default Queue**（默认队列）为识别默认队列。这个属性接受一个整数号码，对应于这个 CQ 概要内一个队列（将用作默认队列）的属性 **Queue Number** 的值。

4. MDRR 概要和 DWRR 概要

配置本地 MDRR 和 DWRR 概要的步骤几乎与配置 CBWFQ 概要的那些步骤相同。本地 PQ 和 CQ 概要在单个概要下包含多个逻辑队列的配置细节，与此不同的是，单个 CBWFQ/MDRR/DWRR 概要在相应调度器内指定单个逻辑队列的配置细节。通过流量策略定义实际的调度器，策略提供 CBWFQ/MDRR/DWRR 概要和流量类之间的映射。

为指定一个本地 MDRR 或 DWRR 概要，需要分别展开 **MDRR Profiles**（MDRR 概要）或 **DWRR Profiles**（DWRR 概要）属性，并设置 **Number of Rows** 属性的值，该属性代表要创建的逻辑队列总数。之后，为每个新建的行，需要配置属性 **Name**（名字）、**Bandwidth Type**（带宽类型）、**Bandwidth Value**（带宽值）、**Priority**（优先级）和 **Queue Limit**（队列限制）。在 DWRR 概要中，属性 **Queue Limit** 被称为 **Queue Limit Value**（队列限制值），且它位于属性 **Queue Configuration**（队列配置）下。这些属性的含义与配置 CBWFQ 概要的相应属性的含义相同。在 DWRR 概要之下存在的其他属性目前没有得到 OPNET 的支持。

5. 监管器概要

本地监管器概要可直接部署在节点接口上，或可用于定义流量策略。为指定一个本地监管器概要，需要执行如下步骤：

1）展开 **Policer Profiles**（监管器概要）属性。

2）设置 Number of Rows 属性的值，该值对应于监管器概要的总数。

3）通过设置如下属性，配置个体概要：

① **Name**（名字）指定监管器概要的名字。

② **Policer Details**（监管器细节）指定当前监治器概要内的监管规则。

注意，每个监管器概要可包含多条监管规则（即单条监管规则对应于 **Policer De-**

tails 属性下的一行)。在概要内的监管规则总数受到属性 **Number of Rows** 的控制。为在概要内配置每条规则，需要指定如下属性：

1) **Match Property**（匹配性质）指定用来识别被监管流量的性质（如到达接口的名字、DHCP 值等）。注意，OPNET 目前不支持使用 QoS Group（QoS 组）和 Rate – Limit Access Control List（速率—限制访问控制列表）性质匹配的流量。

2) **Match Value**（匹配值）用来识别被监管流量的值。这个属性的实际值取决于 Match Property 属性的值。

3) **Bandwidth Type**（带宽类型）控制带宽规格。这个属性仅接受两个值：Absolute（绝对的）和 Relative（相对的），虽然目前 OPNET 仅支持绝对带宽类型。

4) **Average Rate**（平均速率）指定长期平均流量速率。以低于这个值的速率到达的流量被看作是符合约束的。

5) **Peak Rate Information**（峰值速率信息）目前不支持这个属性。

6) **Conform Burst Size (bits)**［符合约束的突发尺寸（比特）］指定流量被看作符合约束的流量突发的最大尺寸（即正常突发尺寸）。

7) **Excess Burst Size (bits)**［过线突发尺寸（比特）］指定流量突发的最大尺寸，高于该值的流量被认为是不符合约束的（即超过速率限制）。在符合约束和过线突发尺寸之间的流量突发，被认为以某个概率是不符合约束的，该概率随流量突发的尺寸增加而增加。

8) **Action Configuration**（动作配置）是一个复合属性，允许对落在速率限制类［Conform（符合约束）、Exceed（过线）（即不符合约束）或 Violate（违规）（即超过符合约束和过线突发尺寸之和）］中的一条报文，指定要实施的动作。与全局监管器的情况一样，可对报文实施两种动作：①改变报文性质［OPNET 目前仅支持改变 DSCP 或 Precedence（优先级）值］；(2) 传输报文［即值 Transmit（传输）］，丢弃报文［即值 Drop（丢弃）］，或将之发送到列表中的下一个监管器（即值 Continue）。如果属性 Action 被设置为 Continue，且为这种流量类型不再有监管器，那么就传输该报文。

6. RED/WRED 概要

可使用属性 **RED/WRED Profiles**（RED/WRED 概要）指定本地 RED 和 WRED 概要。与通常情况一样，概要总数是通过属性 **Number of Rows** 控制的，其中每行代表一个新的 RED 或 WRED 概要。为配置单个 RED 或 WRED 概要，需要设置如下属性：

1) **Name**（名字）为概要的名字。

2) **Match Property**（匹配性质）指定报文性质［如 DSCP、Precedence（优先级）、COS 等］，将被用来在 WRED 机制中区分各种报文类型。

3) **Threshold**（阈值）为一个 RED 或 WRED 机制指定配置参数。RED 机制依赖于配置属性的单个集合（即 min_{th}、max_{th} 和 p_{max}）或由单个 RED 队列组成，其中 WRED 将一个不同的配置参数集合应用到每个流量类（即由多个 RED 队列组成）。属性 **Number of Rows** 控制在这个概要内定义的 RED 队列数量。如果有一个以上的 RED 队列，那么当前概要作为 WRED 工作，否则作为 RED 工作。通过如下属性，配置每个 RED 参数

集合:

① **Match Value**（匹配值）识别将被放入到当前RED队列中的报文。在这个队列中的报文受到当前RED配置的约束。将这个属性的值与通过**Match Property**属性指定的报文性质的值比较，确定报文是否将被放入这个RED队列。

② **Minimum Threshold（packets）**［最小阈值（报文数）］为最小队列占用阈值。

③ **Maximum Threshold（packets）**［最大阈值（报文数）］为最大队列占用阈值。

④ **Mark Probability Denominator**（标记概率标准）为一个最大报文标记概率的标准。

4）**Exponential Weight Constant**（指数权重常数）为估计平均队列占有率的一个常数。

5）**Flow Based**（基于流的）OPNET目前不支持这个功能特征。

这里不描述其他的QoS概要，因为它们不得到OPNET的支持，或配置的功能特征超出了本书的范围。

10.4.3 在一个接口上部署定义好的QoS概要

在一个被仿真网络中部署QoS的最后一步是配置节点接口，使之支持所定义的QoS概要。某些本地概要，如MDRR、DWRR和CBWFQ，不能直接部署在节点接口上。这些概要仅能通过流量策略部署在接口上。为了在节点接口上部署QoS概要，需要展开节点属性**IP... IP QoS Parameters... Interface Information**（IP... IP QoS参数... 接口信息），之后将属性Number of Rows设置为这样一个值，它对应于在节点中接口的总数（将支持的定义好的QoS概要）。接下来，通过设置如图10.28所示的如下属性，需要配置每个个体接口:

1）**Name**（名字）为被配置支持QoS的接口的名字。仅能将这个属性设置为存在于那个节点处的一个活跃接口的名字。这个属性的值字段包含一个下拉菜单，该菜单列出节点中所有的活跃接口。

2）**QoS Scheme**（QoS方案）配置要在这个接口上部署的QoS概要。可在单个接口上部署多个QoS概要。在接口上部署的QoS概要数量是通过属性**Number of Rows**加以控制的。通过如下两个属性，配置每个部署的QoS概要:

① **Type**（类型）指定QoS概要的类型，如Inbound Traffic Policy（进入的流量策略）、Outbound Traffic Policy（外发的流量策略）、Custom Queuing（定制排队）、DWFQ（Class Based）［DWFQ（基于类的）］、DWRR、FIFO、MDRR、MWRR、Priority Queuing（优先级排队）等。

② **Name**（名字）指定所部署的QoS概要的名字。仅可将这个属性设置为在**Type**属性中指定类型的QoS概要。该属性的值字段包含一个下拉菜单，提供所有定义好的QoS概要的一个列表，这些概要是通过**Type**属性所指定类型的概要。

3）**Subinterface Information**（子接口信息）是在这个接口的子接口上，配置QoS概要。

4）**Buffer Size**（**bytes**）［缓冲尺寸（字节）］指定一个接口上缓冲的尺寸。当接口缓冲满时，丢弃所有流量。这个属性定义可由 QoS 概要（部署在这个接口上）的所有逻辑队列使用和共享的缓冲空间总量。必须仔细配置个体逻辑队列，从而使它们总的最大尺寸（以字节数表示）不超过这个属性的值。因为逻辑队列的最大队列尺寸是以报文为单位定义的，所以为了确定以字节表示的最大队列占有率，需要将最大队列尺寸求和，之后以这个接口上的 MTU 尺寸乘以得到的值。

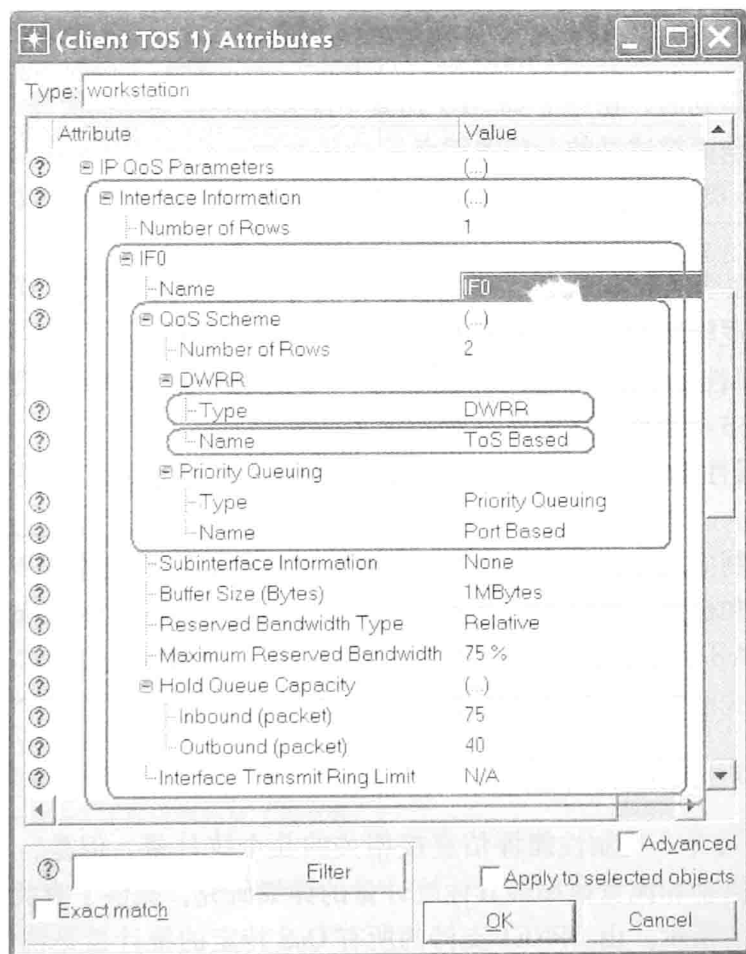


图 10.28　在节点接口上部署 QoS 概要

5）**Reserved Bandwidth**（预留带宽）指定预留带宽类型。这个属性接受如下两个值：

① Relative（相对的）——在属性 **Maximum Reserved Bandwidth**（最大预留带宽）中指定的值，代表链路带宽的一个百分比。

② Absolute（绝对的）——在属性 **Maximum Reserved Bandwidth**（最大预留带宽）中指定的值，代表以比特每秒表示的带宽实际值。

6）**Maximum Reserved Bandwidth**（最大预留带宽）指定 QoS 调度机制（部署在这个接口上）的处理速率（即指定调度器可多快地服务外发流量）。这个值为外发流量

提供可保障带宽的上限。当这个属性被设置为值 N/A 时，整个链路带宽专用于调度器。

7）**Hold Queue Capacity**（保持队列容量）是一个复合属性，指定进入队列（当前不支持）和外发队列的容量。外发队列容量代表接口缓冲的最大容量（以报文数表示），且通过属性 **Hold Queue Capacity... Outbound**（**packets**）[保持队列容量... 外发（报文数）] 进行配置。

8）**Interface Transmit Ring Limit**（接口传输环限制）目前不支持。

10.4.4 结语

正如也许认识到的，在一个 OPNET 仿真中配置和部署 QoS 概要是一项巨大的挑战。在本节，会重复前面描述过的一些关键点：

1）全局 QoS 概要可由场景中的所有节点访问，而本地 QoS 概要仅可由定义它们的节点访问。

2）OPNET 在全局和本地 QoS 概要中支持 LLQ 配置。仅有 CQ、WFQ 和 WRR 调度机制才允许将其逻辑队列之一指定为 LLQ，且每个概要仅有一个 LLQ。

3）通过将属性 **Queue Category**（队列类）设置为 **Low Latency Queue**（低延迟队列）值，全局 QoS 概要将一个逻辑队列指定为一个 LLQ。

4）通过将属性 **Priority**（优先级）设置为 Enabled，本地 QoS 概要将一个逻辑队列指定为一个 LLQ。

通过改变定制应用和标准应用以及流量需求的属性值，可设置所产生报文的 ToS 字段值。所以定制和标准应用以及流量需求的定义，都包含属性 **Type of Service**（服务类型），它指定 ToS 类型的值（即优先级或 DSCP 值）。典型情况下，ToS 值用于流量分类，且它允许显式地将流量断续流组织为不同类。

10.4.5 QoS 相关的统计量

OPNET 提供与 QoS 机制性能评估直接相关的几个统计量。但是，一般而言，对诸如端到端延迟、抖动和流量速率等其他统计量的详细研究，提供了有关各种 QoS 机制性能的一个非常好的指示。由 OPNET 支持的所有 QoS 特定的统计量是位于 IP Interface 类下的节点统计量。为每个接口独立地记录这些统计量。进而，如果一个接口部署这样一种 QoS 机制，它将物理缓冲分成几个逻辑队列（如 WFQ、WRR、CQ 或 PQ），那么为每个逻辑队列单独记录这些统计量。表 10.2 给出所有 QoS 相关统计量的汇总。

表 10.2 在 IP Interface 类中 QoS 相关的节点统计量

名　字	描　述
Buffer Usage（*bytes*）[缓冲使用情况（字节）] *Buffer Usage*（*packets*）[缓冲使用情况（报文数）]	这些统计量以字节和报文数为单位记录在每个接口上队列的尺寸

（续）

名　字	描　述
CAR Incoming Traffic Dropped (*bits/sec*) [丢弃的 CAR 到达流量（比特/秒）] *CAR Incoming Traffic Dropped* (*packets/sec*) [丢弃的 CAR 到达流量（报文数/秒）] *CAR Outgoing Traffic Dropped* (*bits/sec*) [丢弃的 CAR 外发流量（比特/秒）] *CAR Outgoing Traffic Dropped* (*packets/sec*) [丢弃的 CAR 外发流量（报文数/秒）]	这些统计量记录在单个到达或外发接口上由 CAR 流量监管器丢弃的流量总量。典型情况下，流量监管器丢弃不满足约束的报文和超过速率限制的报文。这些统计量是以 bit/s 和报文数/s 为单位记录的
Queue Delay Variation (*sec*) [队列延迟变差（秒）]	这个统计量以秒为单位记录在一个接口的每个队列中报文所经历的延迟变差
Queuing Delay (*sec*) [排队延迟（秒）]	这个统计量以秒为单位记录在一个接口的每个队列中报文所经历的延迟
RED Average Queue Size（RED 平均队列尺寸）	这个统计量记录 RED 机制（部署在一个接口上）估计的针对每个队列的平均队列尺寸
Traffic Dropped (*bits/sec*) [丢弃的流量（比特/秒）] *Traffic Dropped* (*packets/sec*) [丢弃的流量（报文数/秒）] *Traffic Received* (*bits/sec*) [接收的流量（比特/秒）] *Traffic Received* (*packets/sec*) [接收的流量（报文数/秒）] *Traffic Sent* (*bits/sec*) [发送的流量（比特/秒）] *Traffic Sent* (*packets/sec*) [发送的流量（报文数/秒）]	这些统计量记录由一个接口的每个队列丢弃/接收/发送的流量总量。这些统计量是以 bit/s 和报文数/s 记录的

第11章 网络层路由

11.1 引言

路由是计算一个网络中各节点间可能路径的过程。路由是网络层的一项功能，虽然互联网中许多路由协议依赖于传输层协议负责其控制报文的通信。典型情况下，路由是由网络中称为路由器的核心节点实施的，该节点在其路由表中存储路径信息。在给定互联网的巨大尺寸下为保持路由过程是可规模扩展的，互联网被分隔成较小的路由器和网络组。每个这样的分隔由单个权威机构管理，由它定义管理规则和路由策略，并被称为一个 **Autonomous System**（自治系统，AS）。如今在互联网中部署了多样化的路由协议，原因是每个 AS 可独立地决定在其范围内采用哪种路由协议。在一个 AS 内使用的协议称为 AS 内路由协议。另外，AS 间路由协议为 AS 间的数据交付维护路由信息。OPNET 支持许多常见 AS 内协议，包括 RIP、RIPng、OSPF、OSPFv3、EIGRP、IGRP 和 IS - IS。OPNET 也实现 BGP，这是一个 AS 间路由协议。在本章中仅描述两个 AS 内协议，即 RIP 和 OSPF。其他 AS 内协议和 BGP 超出了本书的范围，这里不做讨论。

每个 AS 内路由协议构造要在其域内使用的路由，并以路由表的形式将路由存储在域中的每台路由器内。称为边界路由器的一些路由器可属于多个 AS，那么这些路由器可运行多个路由协议。典型情况下，边界路由器包含多个路由表，它们所属的每个域有一个路由表。所有这些路由表被合并为单个 *IP Forwarding Table*（IP 转发表），该表由网络层内的 IP 进行维护。当一条报文到达一台路由器，要传输到一个特定的目的地时，在 *IP Forwarding Table*（IP 转发表）内 IP 查找那个目的地，以便确定外发接口，这个接口将被指派传输那条报文。

本章后面部分组织如下。本节后面部分专门来讨论在一个网络中如何部署一个特定的路由协议。在 11.2 节和 11.3 节，提供 OPNET 如何对最常用 AS 内路由协议中的两个协议 RIP 和 OSPF 建模的详细描述。这些路由中的第一个协议——**Routing Information Protocol**（路由信息协议，RIP）是运行在 UDP 之上的一个距离向量协议。它仅构造到每个目的地的一条路径，不做任何负载均衡的工作。而 **Open Shortest Path First**（OSPF）（开放最短路径优先）是直接运行在 IP 之上的链路状态协议，进行负载均衡，因为它可构造到每条路径的多条路径。以如下内容的概述结束本章，即可用于路由协议性能评估的 OPNET 统计量以及输出路由信息的步骤，这些路由信息是在一次仿真过程中由被建模网络内各路由器计算得到的。

11.1.1 在一个被仿真网络中部署路由协议

典型情况下，配置一个网络仿真运行期望的路由器，第一步是在网络内部署那些路

由协议。RIP 或 OSPF 都可配置为运行在整个网络上或该网络的一部分上。将一个 *routing domain*（路由域）定义为运行相同路由协议的路由器接口的一个互联集合。可在一个网络的各路由器接口上部署一个或多个路由协议。在核心节点模型中属性 **IP... IP Routing Parameters... Interface Information... <interface name>... Routing Protocol（s）**（IP... IP 路由参数... 接口信息... <接口名>... 路由协议）控制在对应于 <interface name> 的接口上要部署哪些路由协议（即可以是一个以上）。在 **Routing Protocol（s）** 属性的值字段上单击，打开如图 11.1 所示的 **Select Dynamic Routing Protocol（s）**（选择动态路由协议）窗口，可指定在相应接口上应该激活哪些路由协议以及禁止哪些路由协议。

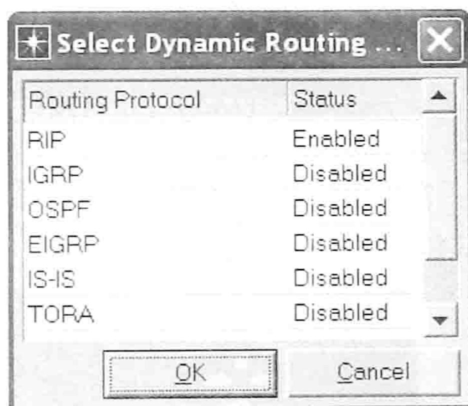


图 11.1　**Select Dynamic Routing Protocol（s）**（选择动态路由协议）窗口

但是，当在一个网络的多个接口上部署路由协议时，一次配置一个接口是一项烦人的和耗时的任务。相反，使用 **Protocols**（协议）菜单选项，在网络中部署期望的路由协议是非常容易的。在一个网络中的接口子集合上部署一个或多个路由协议，步骤如下：

1）选择附接到所关注接口的各链路。通过首先单击一条链路，而后 Shift 单击所有后续链路（即按住 Shift 键并单击所有后续链路）的方法，可选中多条链路。如果希望在网络中所有路由器接口上部署路由协议，则可忽略这一步骤。当一个路由协议部署在一条被选中的链路上时，附接到该链路两侧的接口将被配置运行相应的路由协议。

2）选择 **Protocols→IP→Routing→Configure Routing Protocols**（协议→IP→路由→配置路由协议）选项，这会打开如图 11.2 所示的 **Routing Protocol Configuration**（路由协议配置）窗口。

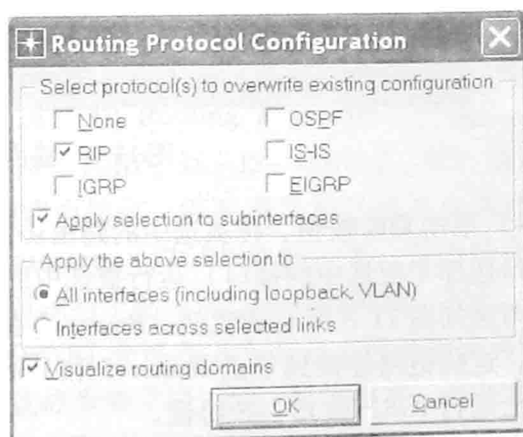


图 11.2　**Routing Protocol Configuration**（路由协议配置）窗口

3）通过指定要部署的路由协议和协议将在其上部署的链路，在网络中配置路由协议部署。注意 **Routing Protocol Configuration** 窗口由三个平板组成：路由协议选择、接口选择和可视化路由域。

① 路由协议选择平板定义将在网络中部署哪些路由协议。通过在路由协议选择平板中单击相应的检查框，可在接口上选择要部署的多个协议。结果是，选择

OSPF将不会取消RIP的默认选择。如果仅希望在被选中接口上部署OSPF，则必须显式地取消RIP的选择框。

② 在接口选择平板中，可指定被选中路由协议是仅部署在被选中的接口上（即附接在被选中链路上的接口）[通过选中单选按钮 *Interface across selected links*（在被选中链路间的接口）] 还是部署在网络中的所有活跃接口上 [通过选中单选按钮 *All interface*（*including loopback*，*VLAN*）（所有接口（包括环回、VLAN））]。

③ 在 *Visualize routing domains*（可视化路由域）检查框中放入一个检查标记，这将产生一个路由协议部署视图，显示带有一个字母的图标，该图标识别被仿真网络中所有链路上的路由协议，如图11.3所示。

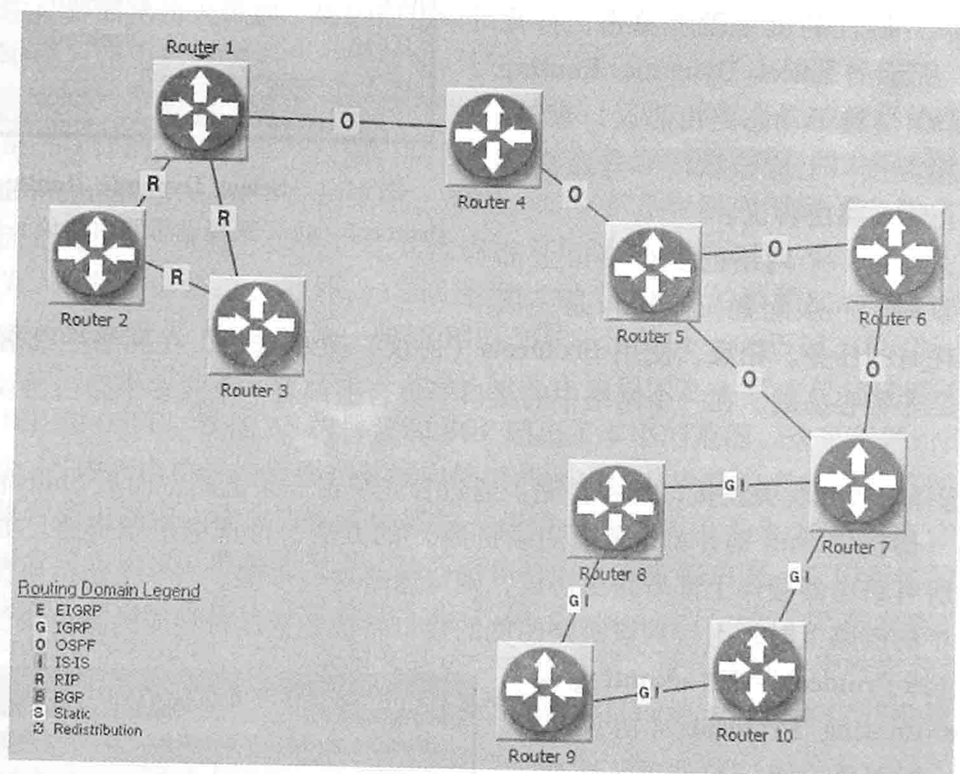


图11.3 路由协议部署范例

4）单击 **OK** 按钮，将被选中的路由协议部署在网络中，这将配置相应接口（即通过接口选择平板选中的接口）运行被选中的路由协议。

考虑如图11.3所示的范例。Router 7在附接到这样的链路的接口上运行OSPF路由协议，这些链路连接到节点Router 5和Router 6，且它在连接Router 8和Router 10的接口上运行IGRP和IS－IS协议。

通过实施如下步骤，可详细研究一台路由器中所有接口上的路由协议部署：

1）右击路由器节点，并选择 **Edit Attributes**（编辑属性）选项。

2）在 **Attributes**（属性）窗口中，展开复合属性 **IP... IP Routing Parameters**（IP... IP路由参数）。

3）单击 **Interface Information**（接口信息）属性的值字段，打开一个窗口，这会

以表的形式显示所有节点接口的配置属性。

图 11.4 显示节点 Router 7 的 **Interface Information** 表。该图显示总共 12 个接口，其中接口 IF4 和 IF5 配置运行两个路由协议 IGRP 和 IS－IS；接口 IF10 和 IF11 运行 OSPF，而所有其他接口配置运行 RIP，这是默认路由协议。

(Interface Information) Table

Name	Status	Operational Status	Address	Subnet Mask	Secondary Address Information	Subinterface Information	Routing Protocol(s)	MTU (bytes)
IF0	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	RIP	Ethernet
IF1	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	RIP	Ethernet
IF2	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	RIP	Ethernet
IF3	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	RIP	Ethernet
IF4	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	IGRP,IS-IS	IP
IF5	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	IGRP,IS-IS	IP
IF6	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	RIP	IP
IF7	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	RIP	IP
IF8	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	RIP	IP
IF9	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	RIP	IP
IF10	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	OSPF	IP
IF11	Active	Infer	Auto Assigned	Auto Assigned	Not Used	None	OSPF	IP

12 Rows　Delete　Insert　Duplicate　Move Up　Move Down

Details　Promote　Show row labels　OK　Cancel

图 11.4　**Interface Information**（接口信息）表

后面将观察到，OPNET 包含名为 **Interface Information** 的几个不同子属性。为避免困惑，将使用不同修饰符指代每个子属性。在本章后面，将称 **IP... IP Routing Parameters... Interface Information**（IP... IP 路由参数... 接口信息）为物理接口特定的属性。RIP 和 OSPF 路由协议的每种协议也都存在称为 **Interface Information** 的其他子属性，将称它们为 RIP/OSPF 接口特定的属性。

11.1.2　配置路由协议属性

经常的情况是，当实施路由协议的仿真研究时，也许需要配置部署在网络中的路由协议。配置个体路由协议的各参数位于节点属性 **IP Routing Protocols**（IP 路由协议）下。如图 11.5 所示，可改变诸如 BGP、EIGRP、IGRP、IS－IS、OSPFv3、RIP 和 RIPng 等路由协议的配置设置。为配置一个特定的路由协议，需要展开相应的复合属性，如 **BGP Parameters**（BGP 参数）、**EIGRP Parameters**（EIGRP 参数）等，并为路由协议配置属性指定期望的值。注意，没有在仿真范围指定路由协议参数的机制。因此，需要一次一个节点地配置路由协议。回顾一下，OPNET 允许在多个节点上同时指定属性值，当在多个节点上指定相同路由协议配置时这就成为凑手的工具。具体而言，当在多个节点上指定一个路由协议配置时，需要实施如下步骤：

1）选择希望指定路由协议配置的所有路由器。

2）在任意被选中的路由器上打开 **Edit Attributes** 窗口。

3）展开属性 **IP Routing Protocols**，之后为期望的路由协议指定配置参数。

4）选中 *Apply to selected objects*（应用到被选中对象）检查框（这将在所有被选中

路由器上复制所引入的改变），并单击 **OK** 按钮保存配置。

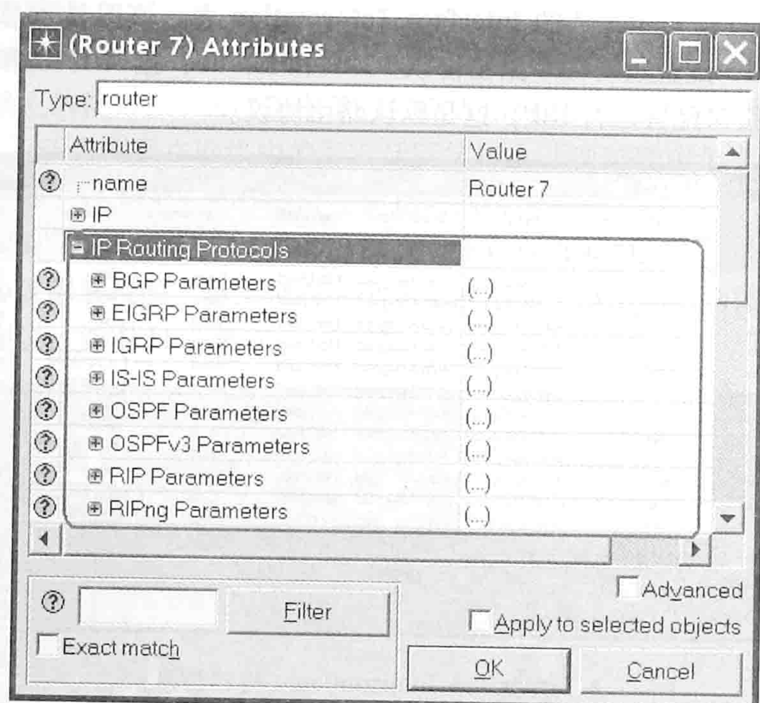


图 11.5 属性 **IP Routing Protocols**（IP 路由协议）

11.2 采用 RIP 进行路由

11.2.1 RIP 简介

RIP 是一种距离向量协议，它使用分布式 Bellman - Ford 算法计算最短路径。每台路由器使用 RIP 与其邻接路由器交换路由信息。每条路由有与之关联的一个度量或成本，这是到目的地的跳数。跳数被限制为 15，跳数 16 被用来表示无穷。OPNET 实现了 RIP 版本 1（RFC 1058）和版本 2（RFC 1723），后者是下面描述的版本；OPNET 也实现了 RIPng，这是用于 IPv6 的 RIP 版本 2 的下一代路由协议。

在 RIP 中，在固定间隔（regular intervals）和网络拓扑发生改变时，每台路由器向其邻居发送路由更新消息。一条路由更新消息包含度量（即典型情况下的跳数），代表从这台路由器到每个目的地网络的成本。当一台路由器接收一次路由更新时，它将更新消息中的每个跳数加 1，并将结果与其为每个目的地而本地存储的值（路由表项）做比较。如果一个目的地的新度量值（接收到的值加 1）小于本地存储值，那么路由器以新值更新那个目的地的路由表项。如果一个目的地的表项被更新，那么那个目的地的下一跳地址被设置为路由更新消息到达的节点地址。如果目的地的度量值等于 16，那么一台路由器认为该目的地是不可达的。除了上述的基本路由更新机制，RIP 提供了各种功能（如在快速变化的网络拓扑时段过程中的稳定性）和抑制（holddown）、水平分割和

毒性反转等机制，防止不正确的路由信息进行传播。

与 RIP 配置相关的所有参数都位于复合属性 **IP Routing Protocols... RIP Parameters**（IP 路由协议 ... RIP 参数）下。如图 11.6 所示，属性 RIP Parameters（RIP 参数）由如下子属性组成：

1）**Process Parameters**（进程参数）包含本地、路由器范围的 RIP 配置参数，这些对这台路由器的所有接口是一样的。在 11.2.2 节描述这些属性。

2）**Interface Information**（接口信息）包含接口特定的 RIP 配置参数。这是一个复合属性，其中总行数等于路由器中物理接口总数。每行包含配置对应于该行之接口的子属性。在 11.2.3 节描述这些属性。同样如前所述，将称这个子属性为 RIP 接口特定的属性，以便将之与 **IP... IP Routing Parameters**（IP... IP 路由参数）之下相同名字的物理接口特定的属性做出区分。

3）**Tunnel Interfaces**（隧道接口）、**VLAN Interfaces**（VLAN 接口）、**BVI Interfaces**（BVI 接口）包含在这台路由器上定义的每个隧道接口、VLAN 接口和 BVI 接口的 RIP 配置参数。这些话题超出了本书的范围，这里不做讨论。

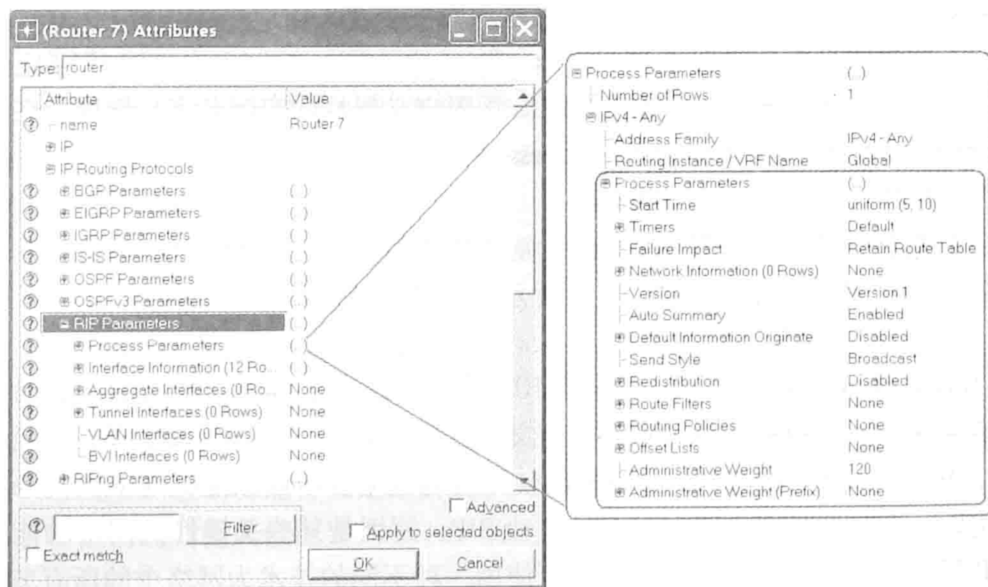


图 11.6 属性 **RIP Parameters**（RIP 参数）

典型情况下，配置 RIP 的过程是十分直接的，并由如下步骤组成：

1）将 IP 地址指派给网络中的路由器接口（即 **Protocols→IP→Addressing→Auto - Assign IP Addresses**（协议→IP→编址→自动指派 IP 地址））。当配置个体接口、调试仿真和检查结果时，显式地在路由器接口上指派 IP 地址有助于解决问题，此时可忽略这个步骤。例如，在网络中已经指派 IP 地址之后，可容易地识别一台路由器中所附接的接口。仅有附接的接口才在 **Interface Information** 表中显式地具有其 IP 地址和掩码值，而所有未附接接口都将这些属性设置为 Auto Assign（自动指派）。

2）在期望的接口上指定 RIP 为路由协议［通过使用下拉选项 **Protocols→IP→Rou-**

ting→Configure Routing Protocols（协议→IP→路由→配置路由协议）]。

3）指定本地路由器范围和接口特定的 RIP 配置参数。也许希望配置这样的 RIP 功能特征，如通告模式、触发的扩展模式、RIP 启动时间、RIP 定时器、RIP 仿真效率模式等。接下来各小节描述这些功能特征。

欲了解有关 OPNET 中所支持 RIP 功能特征的更多信息，请参见产品文档，可通过 **Protocols→RIP→Model User Guide** 菜单选项进行访问。另外，OPNET 软件包括称为 RIP 的一个预配置项目，它提供了各种 RIP 功能特征的绝佳展示，也在 README 场景中包含配置属性的一个简短描述。

11.2.2 本地 RIP 配置属性

如前所述，在一台路由器内对所有接口共同的本地 RIP 参数，可在属性 **IP Routing Protocols... RIP Parameters... Process Parameters**（IP 路由协议 ... RIP 参数 ... 进程参数）之下找到。在这个属性下，子属性 **Number of Rows** 控制这台路由器内不同 RIP 配置定义的总数。默认情况下，在每台路由器中存在一个这样的配置定义，复合属性 **IPv4 – Any** 为 IPv4 单播和组播流量都包含这样的一个定义。

在每个配置定义中，属性 **Address Family**（地址族）控制 RIP 定义将被应用到的流量类型。另外，这个属性的值也确定 RIP 配置定义属性的实际名字。例如，在默认配置定义 **IPv4 – Any** 中，如果将属性 **Address Family** 的值改变为 IPv4 – Unicast，那么父属性的名字也将改变为 **IPv4 – Unicast**。

在每个配置定义内，如 **IPv4 – Any**，复合属性 **Process Parameters** 包含指定当前节点的 RIP 进程配置的参数。如图 11.6 所示的属性 **Process Parameters** 由如下子属性组成：

1）**Start Time**（启动时间）指定当 RIP 进程在这个节点上启动时的仿真时间。这个属性值是通过一个随机概率分布函数定义的。默认情况下，这个属性值被设置为 uniform（5，10），这意味着 RIP 的启动时间使用均匀概率分布函数随机地从区间 5s 到 10s 范围内选取。在一次仿真中不应该太早启动 RIP，原因是某些关键性的网络功能必须在后续 RIP 报文的传输发生之前完成。也可使用一种不同的技术为网络中的所有节点指定 RIP 启动时间，这在 11.2.4 节讨论。

2）**Timers**（定时器）为配置各种 RIP 特定定时器的一个复合属性，这些定时器控制协议的行为。具体而言，可配置如下 RIP 定时器：

① **Update Interval（seconds）**［更新间隔（秒）］确定在邻接路由器之间交换的两次连续路由更新间的时间间隔。这个属性的默认值是 30s。

② **Route Invalid（seconds）**［路由无效（秒）］指定时间段的长度，在此之后将认为路由不再有效。当一条新路由被插入到路由表或只要路由被更新时，就初始化一个路由无效定时器。在这个定时器过期时，路由并不立刻被删除，通过将路由成本设置为无穷而降至标记为无效。这个属性的默认值是 180s。

③ **Flush（seconds）**［清空（秒）］指定时段长度，在此之后将从路由表中清除该

路由。这个定时器的工作方式类似于路由无效定时器的工作方式，例外是当这个定时器过期时，路由被实际删除。路由清空定时器的值应该大于路由无效定时器的值。这个属性的默认值是 240s。

④ **Holdown**（**seconds**）［抑制（秒）］指定抑制定时器的值，从路由无效定时器超期开始启动。在抑制期间，路由器丢弃有关被影响路由的所有更新，以便最小化路由振荡对路由表的影响。OPNET 中目前不支持这个属性，且对仿真性能没有影响。

3）**Failure Impact**（故障影响）在出现一次故障和后续节点恢复时，指定路由器将记录的什么信息。这个属性接受如下两个值：

① Clear Route Table（清除路由表）——配置路由器在出现一次节点故障时清除在这个节点处维护的 RIP 路由表。在节点从故障恢复之后，重建路由表。

② Retain Route Table（保留路由表）——将配置路由器做到当从一次故障恢复时记忆其故障前的 RIP 路由表。这是属性 Failure Impact（故障影响）的默认设置。

4）**Version**（版本）指定路由器作为一个整体所使用的 RIP 版本。版本可以是 Version 1（版本 1）或 Version 2（版本 2）（默认是 Version 1）。个体接口可忽略这个设置，它们可分别使用通过 RIP 接口特定的属性 **Send Version**（发送版本）和 **Receive Version**（接收版本），发送或接收路由更新。

5）**Send Style**（发送风格）指定当一个接口被配置为使用 RIPv1 和 RIPv2 发送路由更新（通过 RIP 接口特定的属性 **Send Version**）时接口将如何处理后向兼容性。这个属性接受两个值：

① Broadcast（广播）——路由器将广播 RIPv2 消息，从而使 RIPv1 路由器可接收路由更新（RFC 2453）。

② Broadcast & Multicast（广播和组播）——路由器将广播 RIPv1 消息，并组播 RIPv2 消息。

6）**Auto Summary**（自动汇总）控制 RIP 是否实施 RFC 1058 中定义的子网过滤。默认情况下，这个功能特征是激活的。

路由器范围的其他属性配置高级 RIP 功能特征，在本书中不做讨论。

11.2.3　RIP 接口特定的配置属性

为给一个接口指定 RIP 接口特定的配置属性，需要设置复合属性 **IP Routing Protocols... RIP Parameters... Interface Information... <interface name>**（IP 路由协议... RIP 参数... 接口信息... <接口名>），其中 **<interface name>** 识别关注的一个接口。如图 11.7 所示，RIP 接口特定的配置包括如下属性：

1）**Name**（名字）指定接口名字。

2）**Status**（状态）指定 RIP 是否运行在这个接口上。但是，这个属性对仿真执行没有影响。在这个层次上不能修改这个属性的值，原因是其值的设置依据路由协议是如何配置运行在每个节点的各种接口上的。

3）**Silent Mode**（静默模式）指定 RIP 如何运行在这个接口上，且接受两个可

能值：

① Disabled（禁止）——这个接口发送和接收 RIP 更新。

② Enabled（激活）——虽然这个接口仍然接收和处理 RIP 更新，但它不能发出 RIP 更新。

4）**Cost**（成本）指定与这个接口上接收到的 RIP 更新相关联的成本。所有接口的默认成本是 1。对于在一次更新中接收到的每个目的地，将这个成本加到其度量值。

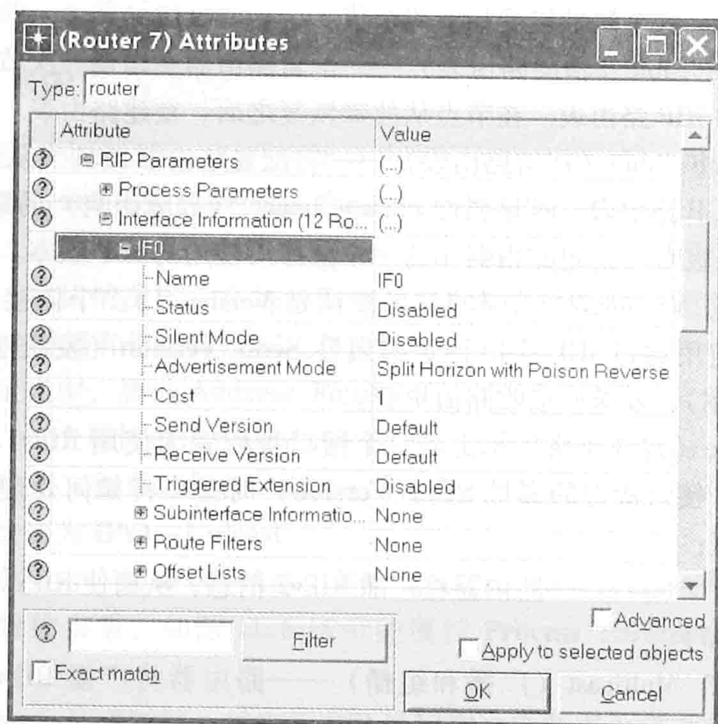


图 11.7 RIP 接口特定的配置参数

5）**Send Version**（发送版本）指定用来发送更新消息的 RIP 版本。这个属性的值覆盖路由器范围属性 **Version** 的值。这个属性接受如下值：

① Version1——通过广播发送 RIPv1 更新。

② Version2——通过组播发送 RIPv2 更新。

③ Version1&2——路由器可依据路由器特定属性 **Send Style** 的设置，发送 RIPv1 和 RIPv2 的更新消息。

④ Default——取决于通过路由器范围属性 Version 指定的 RIP 版本，使用 RIPv1 或 RIPv2 更新方法。

6）**Receive Version**（接收版本）指定由这个接口接受的 RIP 路由更新类型。这个属性可被设置为如下值之一：

① Version1——仅接受 RIPv1 更新。

② Version2——仅接受 RIPv2 更新。

③ Version1&2——接受 RIPv1 和 RIPv2 更新。

④ Default——如果路由器范围 RIP 配置属性 Version 设置为 Version 1，那么接受 RIPv1 和 RIPv2 更新，否则（即 Version 2）仅接受 RIPv2 更新。

7）**Advertisement Mode**（通告模式）指定在当前接口上如何实施 RIP 通告。这个属性接受如下三个值：

① No Filtering（没有过滤）——路由器通过在这个 IP 接口上的所有路由表项，包括通过在这个接口上接收到的通告学习到的那些表项。

② Split Horizon（水平分割）——路由器通过在这个 IP 接口上的所有路由表项，但不包括通过在这个接口上接收到的通告学习到的那些表项。**split horizon advertisement mode**（水平分割通告模式）降低了 **count to infinity problem**（计数到无穷的问题），这个问题折磨着所有距离向量协议，如 RIP。

③ Split Horizon with Poison Reverse（带有毒性反转的水平分割）——路由器通告在这个接口上的所有路由表项，而通过这个接口上接收到的通告而学习到的路由，则以一个无穷的度量（即 16）加以通告。**Poison reverse**（毒性反转）是水平分割模式的增强措施，有助于加速路由收敛。

8）**Triggered Extension**（触发扩展）属性指定在这个接口上激活或禁止 **triggered extension mode**（触发扩展模式）。这个属性仅接受两个值：Enabled 和 Disabled。如果激活触发扩展模式，则在那个接口上不发送正常的周期性更新。仅当接口初始化和这个路由器中的路由表经历一次变化时，才发送路由更新。具体而言，在发生一次路由表变化之后，在使用均匀概率分布函数（输出处于 1 ~ 5s 的区间范围内）计算的一个时间间隔之后，在接口上发送一次路由更新。如果对一个接口禁止触发扩展模式，那么依据更新间隔定时器的值，即本地路由器范围 RIP 属性 **Update Interval**（**seconds**）[更新间隔（秒）] 周期性地发送路由更新。注意为了使触发扩展模式正常工作，在同一子网内的所有接口上都必须激活 Triggered Extension 属性。

9）**Subinterface Information**（子接口信息）、**Route Filters**（路由过滤器）和 **Offset Lists**（偏移列表）描述各种高级 RIP 功能特征，本书中不讨论。

11.2.4 配置 RIP 启动时间

为 RIP 指定启动时间有两种方式：在个体路由器上设置 **Start Time** 属性的值（11.2.2 节）或通过使用 **Protocols** 菜单。相比于直接改变节点属性的值，第二种方法是配置 RIP 启动时间的一种比较方便的方式，原因是这种方法可同时应用到一个路由器集合。为通过 **Protocols** 菜单指定 RIP 启动时间，需要实施如下步骤：

1）在希望指定 RIP 启动时间的网络中，选择支持 RIP 的路由器。如果希望在网络中的所有路由器上配置 RIP 启动时间，则可忽略这一步骤。

2）选择 **Protocols→RIP→Configure Start Time...**（协议→RIP→配置启动时间...）选项，打开如图 11.8 所示的 **RIP Start Time Configuration**（RIP 启动时间配置）窗口。

3）从分布的下拉列表中选择将被用来计算 RIP 启动时间的概率分布函数的名字。

4）在 *Mean outcome*（均值输出）文本框为概率分布函数指定一个输入参数。如图 11.8 所示，constant（常数）概率分布函数带有单个参数。其他分布函数，如 uniform（均匀分布），可带有一个以上的参数，其中将出现额外文本框。

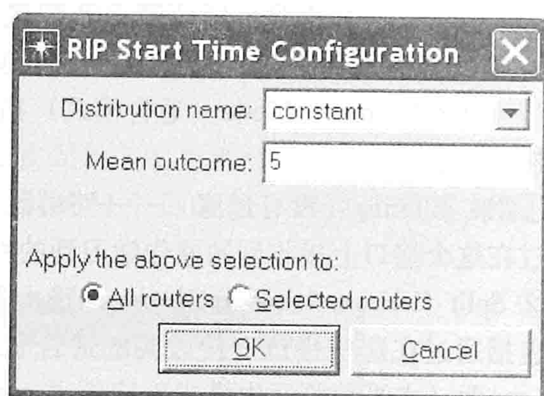


图 11.8 **RIP Start Time Configuration**（RIP 启动时间配置）窗口

5）选中相应的单选按钮，指定希望将 RIP 启动时间配置应用到所有路由器，还是仅应用到在第一步中选择的路由器。

6）单击 **OK** 按钮应用配置，并关闭窗口，在被选中路由器（即网络中的所有路由器或第一步中选中的那些路由器）上这将本地路由器范围属性 **Start Time** 设置为指定的分布函数和均值输出。

11.2.5 RIP 仿真效率模式

为改善仿真效率，OPNET 支持一个属性配置集合，这些配置将加快仿真执行。例如，在不经历拓扑变化的一个网络中，在某个时间段（即当路由表收敛到一个稳定状态时）之后继续交换路由更新消息可能是不必要的。出于这个目的，OPNET 包含两个全局 RIP 特定的属性，在某段时间之后，防止路由器通告路由更新：

1）**RIP Sim Efficiency**（RIP 仿真效率）指定 RIP 仿真效率模式是激活的还是禁止的一个属性。当设置为 Enabled 时，在仿真过去通过 **RIP Stop Time**（**seconds**）［RIP 停止时间（秒）］属性配置的时间值之后，网络中的路由器将停止发送 RIP 更新。在 RIP 不是研究对象的多数场景中，通过降低必须由仿真内核处理的事件数量，对这些更新的抑制可改进仿真效率。当设置为 Disabled 时，在整个仿真期间，路由器将继续发送 RIP 路由更新。当仿真中的网络经历拓扑变化或当 RIP 是研究的焦点时，这种设置是有用的。

2）**RIP Stop Time**（**seconds**）［RIP 停止时间（秒）］指定在整个时间之后在网络中将不再产生 RIP 路由更新。仅当 **RIP Sim Efficiency** 设置为 Enabled 时，这个属性才是适用的。默认情况下，OPNET 将 **RIP Stop Time**（**seconds**）设置为 65s。

如图 11.9 所示，可通过 **Inputs... Global Attributes... Simulation Efficiency**（输入...全局属性...仿真效率）复合属性配置这些参数，该属性可通过 **Configure/Run DES** 窗口访问。通过选择 **DES→Configure/Run Discrete Event Simulation...**（DES→配置/运行离散事件仿真）选项，按 Ctrl + R 键，或通过单击 Running Man（奔跑的人）图标（见 4.3 节），可打开这个窗口。

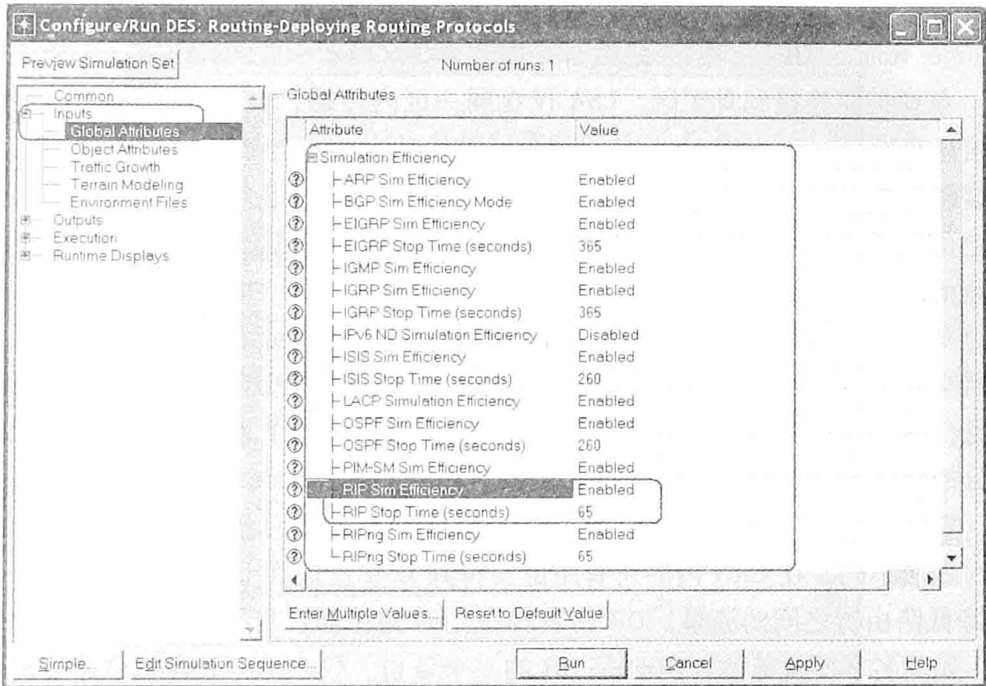


图 11.9　指定 RIP 仿真效率模式

11.3　采用 OSPF 路由

11.3.1　OSPF 简介

开放最短路径优先（Open Shortest Path First，OSPF）是一种链路状态路由协议。在 OSPF 中，每台路由器向其域中的所有其他路由器发送链路状态通告（Link - state Advevtisement，LSA）。接收到的链路状态信息在每台路由器上作为一个链路状态数据库（Link - state Database，LSDB）加以维护，该数据库本质上是整个网络拓扑的一种表示。在接收到链路状态信息时，每台路由器使用 Dijkstra 的最短路径优先（SPF）算法独立地计算到所有目的地的最短路径。通过一种修正的洪泛算法完成 LSA 的传输，该算法快速地将 LSA 传播到所有路由器，但剪枝重复的更新。结果，相比于 RIP，OSPF 快速地检测到拓扑中的变化（如链路失效）并非常快速地收敛到相对无环路的路由。

OSPF 将一个外部指定的成本度量指标与每条链路关联。成本可表示链路带宽或任何其他期望的参数，如距离、延迟、可靠性等。当存在到一个目的地的等成本的多条路径时，OSPF 在其路由表和 *IP Forwarding Table*（IP 转发表）中包括所有这些路径。这允许 IP 在到一个给定目的地的等成本路径间进行负载均衡。

一个 OSPF 路由域可被分为几个区。典型情况下，每个区是连续网络和附接主机的一个群组。以 32 比特整数识别各区，该整数表示为一个简单的十进制数或类似于 IPv4 地址、基于字节的点分十进制表示法。依据惯例，区 0（或 0.0.0.0）代表核心或骨干

区，在所有其他区之间提供连通性。属于多个区的一台路由器被称为一台区边界路由器（Area Border Router ABR），但该路由器的每个接口仅属于一个区。所有区必须通过一台 ABR 或一条虚链路连接到骨干区。LSA 仅在同一区内传播，但一台 ABR 参与它所属每个区的信息交换。一台 ABR 也为它的每个区维护独立的拓扑数据库。有时在区边界处的路由信息可被压缩精简（condensed），方法是将一组路由组合为单条通告，这降低了路由器上的负载和所感知的网络复杂性。将路由归组的这个过程被称为路由汇聚（route aggregation）。

因为 OSPF 是一种链路状态路由协议，所以 OSPF 路由器建立邻居关系，以便与其他路由器交换路由更新。路由器使用 Hello 报文发现和维护邻居关系。Hello 消息也作为保活消息，将一台路由器仍然功能正常的信息通知邻居。作为一个 LAN（如以太网）组成部分的路由器，选择一台指代的路由器（Designated Router，DR）和一台备份指代路由器（BDR）。DR/BDR 节点的主要目的是降低 LSA 流量总量。与相互间交换链路状态更新的做法不同，在 LAN 内的所有路由器将其 LSA 发送到 DR，DR 作为一个集线器（hub）降低路由器之间的流量。DR 代表所有 LAN 路由器，与同一区内的其他路由器交换 LSA。如其名字所意味的，BDR 是 DR 的一个备份，仅当 DR 下线时它才发送链路状态更新。

在一台路由器上可配置多个 OSPF 进程。各进程使您可在同一节点的不同接口上运行不同的 OSPF 配置。一台路由器的各接口可被配置为，运行在那台路由器上配置的各进程中的一个进程。虽然可从其他 OSPF 进程学习到路由，但每个 OSPF 进程构建其自己的 LSDB。另外，OSPF 和 RIP 可被配置为，通过接收由其他路由协议（配置在该路由器上）学习到的路由而构造其路由表。这项技术称作路由再发布，在本书中不做讨论。

OPNET 支持 IPv4 的 OSPF 版本 2（RFC 2328）和 IPv6 的 OSPF 版本 3（RFC 5340）。在本章中，描述 OSPF 的默认版本，即版本 2，在本书中不讨论 OSPF 版本 3。

11.3.2 OSPF 属性

与 OSPF 配置有关的所有参数都位于复合属性 **IP Routing Protocols... OSPF Parameters**（IP 路由协议 ... OSPF 参数）下。如图 11.10 所示，属性 OSPF Parameters 由如下子属性组成：

1）**Processes**（进程）指定运行在这台路由器中各 OSPF 进程的配置。

2）**Interface Information**（接口信息）指定在这台路由器内每个 IP 接口上的 OSPF 配置。同样，与 RIP 的情况相同，将称这个属性为 OSPF 接口特定的，以便将之与 **IP... IP Routing Parameters**（IP... IP 路由参数）下相同名字的物理接口特定的属性相区分。

3）**Aggregate Interfaces**（汇聚接口）、**Loopback Interfaces**（环回接口）、**Tunnel Interfaces**（隧道接口）、**VLAN Interfaces**（VLAN 接口）和 **BVI Interfaces**（BVI 接口）在各种其他类型的接口上指定 OSPF 配置。在本书中不讨论这些属性。

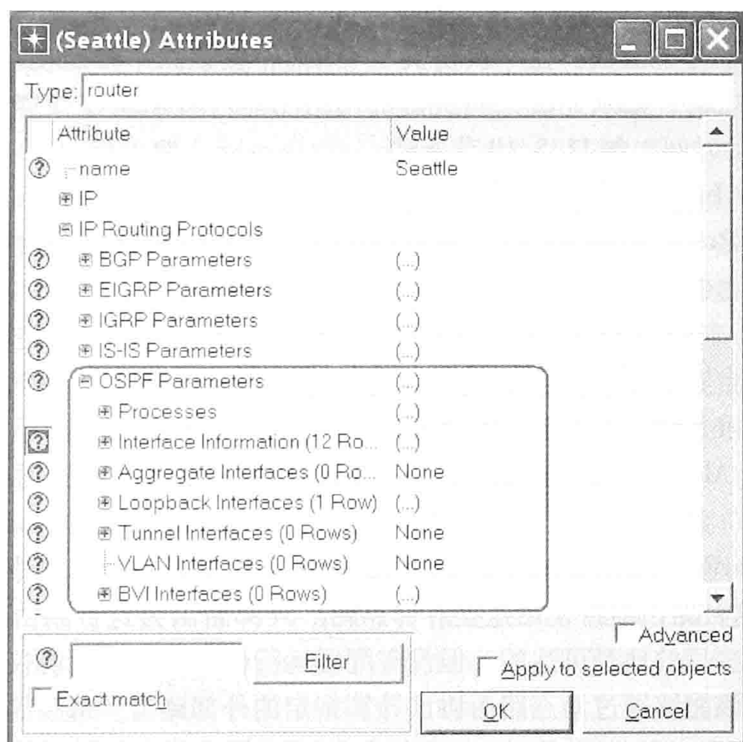


图 11.10　属性 **OSPF Parameters**

11.3.3　配置 OSPF 进程

为配置个体 OSPF 进程，需要展开属性 **IP Routing Protocols... OSPF Parameters... Processes**（IP 路由协议 ... OSPF 参数 ... 进程），并设置属性 **Number of Rows** 为希望配置的 OSPF 进程总数。默认情况下，OPNET 将属性 **Number of Rows** 设置为 1。每个 OSPF 进程是通过如下参数指定的：

1）**Process Tag**（进程标签）指定唯一识别这个 OSPF 进程的一个字符串。

2）**VRF Name**（VRF 名字）为与这个进程相关联的 VPN 路由和转发（VRF）表的名字。在本书中不讨论这个属性。

3）**Process Parameters**（进程参数）为 OSPF 进程的配置参数。

4）**Address Family**（地址族）是 OSPF 为之提供其路由服务的流量类型。默认情况下，OSPF 配置为 IPv4 单播流量提供路由服务。

这里主要关注 **Process Parameters** 复合属性，它指定 OSPF 进程配置。如图 11.11 所示，**Process Parameters** 由如下子属性组成：

1）**Router ID**（路由器 ID）指定一个唯一的 OSPF 路由器识别符，可写为一个正整数值或表示为点分半字节（dotted quad）表示法。如果 **Router ID** 被设置为 Auto Assigned，那么在 **IP... IP Routing Parameters... Router ID**（IP... IP 路由参数 ... 路由器 ID）下指定的值被用作 OSPF 路由器识别符。

2）**Start Time**（启动时间）指定在这个节点上OSPF启动的时间。这个属性的值被指定为一个概率分布函数。在一次仿真中OSPF不应该太早地启动，原因是在OSPF报文可成功传输之前，必须完成某些关键的网络功能。OPNET也使您通过Protocols菜单指定OSPF启动时间，如11.3.10节所述。

3）**Network Information**（网络信息）属性对仿真没有影响。

4）**Default Route Information**（默认的路由信息）指定这台路由器是否将自己通告为一台默认路由器。目前OPNET仅支持两个可能的设置：

① Not Used（未使用的）和Originate（源发的）——这两个值将配置路由器不将其自己通告为一台默认路由器。这两种设置的含义是不同的，但OPNET不支持Originate设置，这实际上导致两者相同的行为。

② Originate Always（总是源发的）——路由器将总是将其自身通告为一台默认路由器。

5）**External Route Information**（外部路由信息）指定一个静态外部路由列表，在OSPF路由域内这台路由器将使用这些外部路由。外部路由为这台路由器提供到目的地的路径信息，这些目的地是可达的，但没有配置运行任何动态路由协议。注意，在网络中的路由器不应该能够通过动态路由协议计算指定的外部路由。每条静态外部路由表项是作为External Route Information表中一个独立行指定的，且由标准路由信息组成，这些路由信息是通过如下属性配置的：

① **Destination Address**（目的地地址）和**Destination Mask**（目的地掩码）为这个路由表项指定一个目的地地址范围。

② **Cost**（成本）代表到达这些目的地地址的成本。

③ **Metric Type**（度量类型）指定如何解释属性**Cost**的值。这个属性接受两个值：Type 1（类型1）（意味着路径成本是作为**Cost**属性值和到达中间路由器的成本这两者的和计算的）和Type 2（类型2）（意味着**Cost**属性值由其本身使用计算路径成本）。

④ **Forwarding Address**（转发地址）指定到目的地的路径上的下一跳地址。

6）**Area Information**（区信息）指定为这台路由器配置的区。虽然在这个属性中的区信息是可编辑的，但一般来说，这样做不是一个好想法，因为一些信息是难以得到的（猜出的）。相反，应该使用11.3.8节中描述的**Protocols**菜单选项配置各区。**Area Information**是一个复合属性，为区信息的每个定义包含一个独立的行。每行包含如下信息，如**Area ID**（区ID）、**Area Type**（区类型）、**Default Route Information**（默认路由信息）以及一些其他高级属性。

7）**Area Summarization**（区汇总）指定为每个连接的区，这台路由器通告或隐藏的地址范围。在11.3.9节描述如何为边界路由器配置区汇总。

8）**Address Summarization**（地址汇总）为OSPF创建汇聚地址。这个复合属性使每行代表单个地址汇聚，这是通过如下属性定义的：

① **Address**（地址）和**Subnet Mask**（子网掩码）定义地址范围，将代表单个地址汇聚。

② **Advertise**（通告）指定路由器是否通告匹配所定义地址汇聚的各路由（即当这个属性设置为 Disabled 时，将不通告路由）。

③ **Tag**（标签）为通过路由映射控制再发布指定一个“匹配”值。

这个属性对 Cisco 路由器上 **summary - address** 命令进行建模。

9）**SPF Calculation Parameters**（SPF 计算参数）通过三个属性配置最短路径优先（SPF）算法的计算。属性 **Style**（风格）指定将如何实施 SPF 计算。如果设置为 Periodic（周期性的），则路由器周期性地实施计算，前提条件是由于 LSDB 中的变化，存在一次计算的需要。如果 **Style** 设置为 LSA Driven（LSA 驱动的），那么当接收到一条 LSA，但在等待由属性 **Delay**（**seconds**）[延迟（秒）] 指定的时段之后，路由器实施计算。在这两种情形中，两次连续 SPF 计算必须相隔至少由属性 **Hold Time**（**seconds**）[保持时间（秒）] 指定的时间量。

10）**Reference Bandwidth**（参考带宽）指定用来计算 OSPF 接口成本的带宽值。具体而言，如果 OSPF 接口特定属性 **Interface Information... < interface name > ... Cost**（接口信息...<接口名>... 成本）[指定在那个接口上发送一条报文的成本（即接口成本）] 被设置为 Auto Calculate（自动计算），那么接口成本计算为 **Reference Bandwidth** 属性的值和附接于这个接口的链路上的可用带宽之比。这种计算方法将一个较低的接口成本指派给具有较高带宽的链路。因此，如果属性 Reference Bandwidth 设置为 100Mbit/s，那么一个具有带宽为 10Mbit/s 的接口将有接口成本 10，而具有带宽为 1Mbit/s 的一个接口将有成本 100。

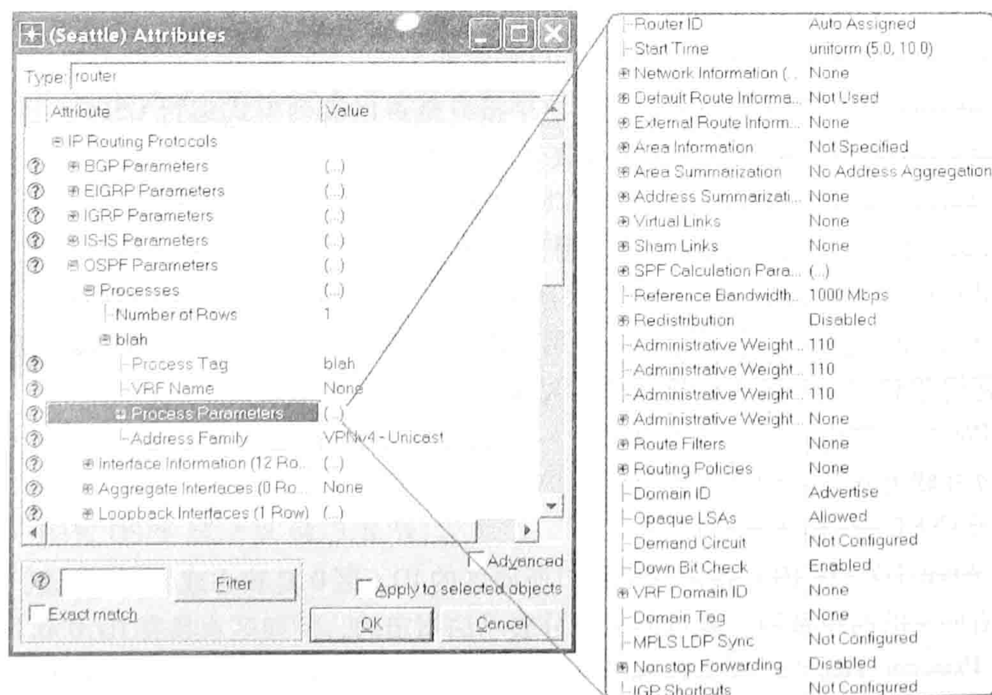


图 11.11　OSPF 配置属性 **Process Parameters**（过程参数）

配置一个OSPF过程的其他属性，指定各种高级功能特征，这些不在本书中讨论。欲了解有关可用OSPF过程配置参数的更多信息，参见OPNET文档［可通过 **Protocols→OSPF→Model User Guide**（协议→OSPF→模型用户指南）选项进行访问］和个体 **Process Parameters**（进程参数）属性的描述（可通过位于每个属性左侧带有问号图标的蓝色圆圈进行访问）。也会发现阅读 *Cisco's OSPF Design Guide*（Cisco OSPF设计指南）（即文档ID：7039）和相关的Cisco文档是有用的，看来这些文档经常被用作OPNET的OSPF实现的参考指南。

11.3.4 在路由器接口上指定OSPF配置

OSPF接口特定属性 **Interface Information**（接口信息）［即 **IP... OSPF Parameters... Interface Information**（IP... OSPF 参数... 接口信息）］在一个路由器的接口上指定OSPF配置。事实上，每行包含在这台路由器上单个物理接口的OSPF配置，且每行的名字是相应物理接口的名字。通过图11.12设置如下属性的值，可在一个接口上指定OSPF配置：

1）**Name**（名字）是一个只读属性，指定正被配置的物理接口的名字。

2）**Status**（状态）指定在这个接口上是激活还是禁止OSPF。不能直接修改该属性：其值已经基于部署在这个接口上的路由协议进行了设置。如果这个接口配置为运行OSPF，那么 **Status** 被设置为Enabled，否则设置为Disabled。回顾一下，部署在一个接口上的路由协议，是通过物理接口特定属性 **IP... IP Routing Parameters... Interface Information... <interface name>... Routing Protocol(s)**（IP... IP路由参数... 接口信息... <接口名>... 路由协议）加以指定的。

3）**Silent Mode**（静默模式）指定这个接口是否以被动模式运行OSPF。当这个属性设置为Enabled时，将不会发送或接收任何LSA。

4）**Type**（类型）指定一个底层接口的类型。这个属性接受如下值：

① Point To Point（点到点）——用于SLIP链路。

② Broadcast（广播）——用于以太网和具有广播能力的其他类型的接口。

③ Non-Broadcast（非广播）——用于没有广播能力的多接入接口，如ATM或帧中继，它们具有以全互联方式建立的永久虚电路（PVC）。

④ Point to Multipoint（点到多点）——用于没有广播能力的多接入接口，它们具有以部分互联方式建立的永久虚电路（PVC）。

⑤ MANET——用于无线接口。

5）**Area ID**（区ID）指定这个接口所属区的ID。区0是核心或骨干区。区ID可以是一个诸如4的简单整数，或以四数点分表示法指定的，例如区4具有ID 0.0.0.4。

6）**Process Tag(s)**（进程标签）指定配置哪个OSPF进程运行在这个接口上。如果这个属性被设置为多个值，那么仅考虑列表中的第一个值。这是一个本地OSPF进程标签，因此它必须匹配其邻接节点的那些标签。

7）**Cost**（成本）指定在这个接口上发送一条报文的成本。可指定成本值为一个整

数或将之设置为 Auto Calculate（自动计算）。当设置为 Auto Calculate 时，基于附接链路的带宽和属性 **Reference Bandwidth** 的值，像在 11.3.3 节描述的那样计算成本值。在这种方法中，低带宽链路被指派一个较高的成本，而高带宽链路被指派一个较低的成本。在 11.3.6 节，描述为所有或选中接口配置链路成本的另一种方法。

8）**Timers**（定时器）是为这个接口指定各种 OSPF 特定的定时器。这些在 11.3.7 节加以描述。

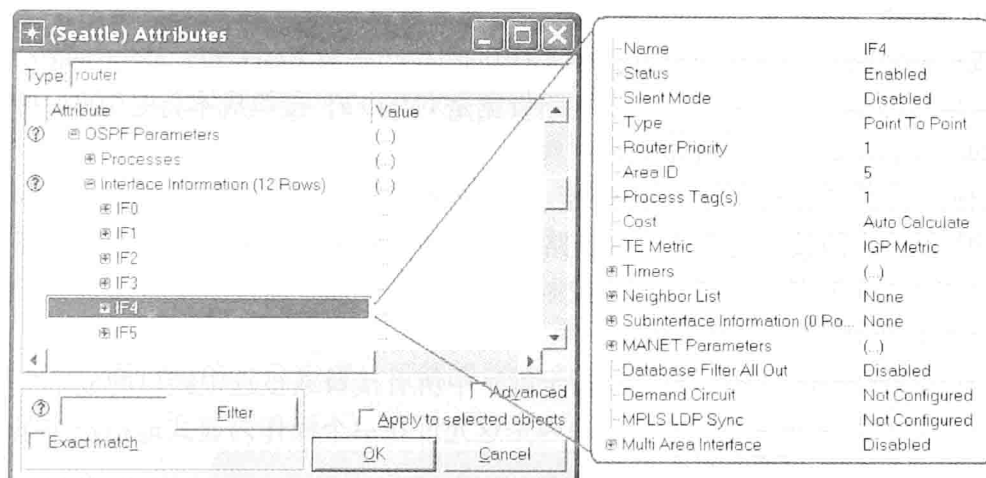


图 11.12　在接口上配置 OSPF 的属性

11.3.5　配置 OSPF

使用 OSPF 的最简单方式是将之留在其默认模式。当然，必须像在 11.1.1 节中描述的那样在期望的路由器接口上部署 OSPF，原因是 IPv4 的默认路由协议是 RIP。当以其默认模式部署 OSPF 时，存在单个区，所有节点属于这个默认的号为 0 的骨干区。同样，在每个节点上仅运行一个 OSPF 进程。可使用如下步骤，为一个比较复杂的运行环境配置 OSPF：

1）为所有路由器接口指派 IP 地址（见 9.3 节）。

2）在期望的路由器接口上部署 OSPF（见 11.1.1 节）。

3）配置链路成本（见 11.3.6 节）。

4）配置 OSPF 定时器（见 11.3.7 节）。

5）配置 OSPF 区（见 11.3.8 节）。

6）配置 ABR（见 11.3.9 节）。

7）配置启动时间（见 11.3.10 节）。

8）配置仿真效率模式（见 11.3.11 节）。

通过使用 **Protocols→OSPF**（协议→OSPF）选项，可实施这些配置步骤中的一些步骤，该选项提供了可能 OSPF 配置操作的一个菜单。使用这项技术的优势是，这些操作中的多数操作可在单个步骤内实施于所有路由器接口或被选中的接口。

11.3.6 为 OSPF 路由配置链路成本

链路成本（或接口成本）是针对每个接口指定的，并用作最短路径计算的基础。接口成本代表通过这个接口路由一条外发报文的成本。进入的（inbound）成本是没有指定的，并被链路另一侧看作外发成本。接口成本可被指定为一个整数值或依据参考带宽和链路带宽值计算得到。如在 11.3.3 节所述，每个 OSPF 进程包含配置属性 **Reference Bandwidth**（参考带宽），该属性默认情况下被设置为 100Mbit/s。链路带宽取决于附接到接口的链路类型。在仿真过程中，接口成本计算为 **Reference Bandwidth** 属性和链路带宽值之间的比率。因此，如果链路带宽是 1Mbit/s，接口成本将是 100。可使用下面描述的四种方法之一指定链路成本：

1）使用配置链路成本的默认设置。在这样一种情形中，仿真将依据 **Reference Bandwidth** 属性和附接于这个接口的链路的带宽（链路带宽）的值，自动地计算链路成本。这种方法的劣势是，不能显式地控制链路成本。相反，链路成本直接依赖于在相应链路上分配的带宽。

2）使用 **Protocols** 菜单，显式地指定网络中所有接口或被选中接口的成本。这是指派链路成本的最简单和首选的方法，原因是这允许在单个操作内显式地指定多条链路的成本。为在多条链路上指定链路成本，需要实施如下步骤：

① 选择希望以相同成本值配置的各链路。如果希望将网络中所有连接的接口上的链路成本设置为相同值，则可忽略这一步骤。

② 选择 **Protocols→OSPF→Configure Interface Cost**（协议→OSPF→配置接口成本）菜单选项，打开如图 11.13 所示的 **OSPF Interface Cost Configuration** 窗口。

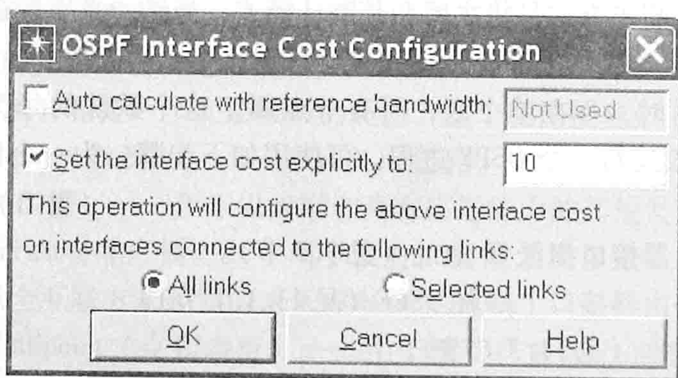


图 11.13 **OSPF Interface Cost Configuration**（OSPF 接口成本配置）窗口

③ 选择检查框 *Set the interface cost explicitly to*（显式地将接口成本设置为），并输入期望的成本值。

④ 仅分别选中 *All links*（所有链路）或 *Selected links*（被选中的链路）上的单选按钮，就可指定希望在所有链路还是被选中链路上指定成本配置。

⑤ 单击 **OK** 按钮，完成这项操作。

⑥ 对可能要求一个不同成本值的其他链路，可重复这个过程。

3）通过配置参考带宽和链路带宽值，指定链路成本。为网络中所有链路设置链路带宽的第一步，可如下做到这一点：

① 选择希望以相同链路带宽值配置的各链路。如果希望将网络中所有连接的接口上的链路带宽设置为相同值，则可忽略这一步骤。

② 选择 **Protocols→IP→Routing→Configure Interface Metric Information**（协议→IP→路由→配置接口度量信息），打开如图 11.14 所示的 **Configure Interface Metric Information**（配置接口度量信息）窗口。

③ 将文本框 *Bandwidth*（*kbps*）的值设置为期望的链路带宽。注意，这里的带宽单位是 kbit/s，所以如果想要 10Mbit/s 的带宽，必须输入 10 000。

④ 仅分别选中 *All connected interfaces*（所有连接的接口）或 *Interfaces across Selected links*（被选中的链路间的各接口）单选按钮，就可指定希望在所有连接的各接口还是被选中链路的各接口上指定链路带宽。

⑤ 单击 **OK** 按钮，完成这项操作。

⑥ 对可能要求一个不同链路带宽值的其他链路，可重复这个过程。

接下来，通过实施如下步骤，可能想将参考带宽设置为不同于其默认值 100Mbit/s 的一个值：

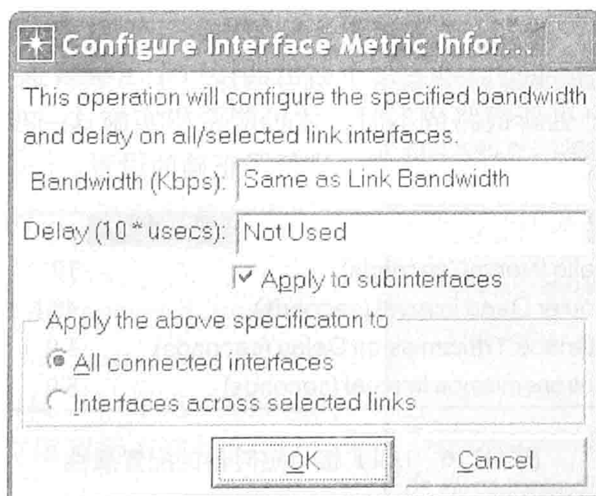


图 11.14　**Configure Interface Metric Information**（配置接口度量信息）窗口

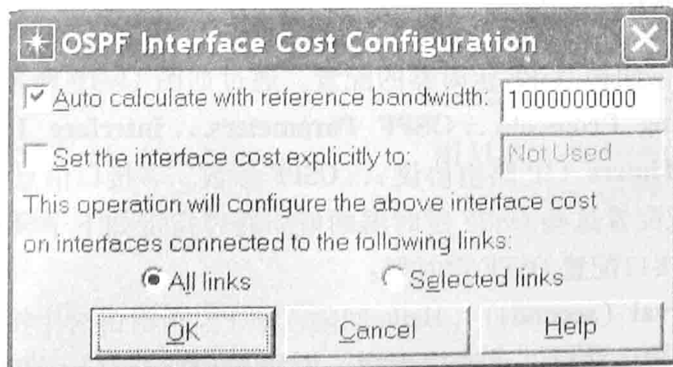


图 11.15　**OSPF Interface Cost Configuration**（OSPF 接口成本配置）窗口

① 选择希望以相同参考带宽值配置的各链路。如果希望将在所有连接接口上的参考带宽设置为相同值，则可忽略这一步骤。

② 选择 **Protocols→OSPF→Configure Interface Cost option**（协议→OSPF→配置接口成本选项），打开如图 11.15 所示的 **OSPF Interface Cost Configuration**（OSPF 接口成本配置）窗口。

③ 选择检查框 *Auto calculate with reference bandwidth*（以参考带宽自动计算），并在相邻文本框中输入参考带宽的期望值。注意，文本框中提前带有数 1 000 000 000，且单位是 bit/s。这个值对应于 1.0Gbit/s，这与默认值不同。

④ 仅分别选中 *All links*（所有链路）或 *Selected links*（被选中的链路）上的单选按钮，就可指定希望在所有连接的各接口还是被选中链路的各接口上指定参考链路带宽。一般而言，将所有网络链路上的参考带宽设置为相同值，是一种不错的想法。

⑤ 单击 **OK** 按钮，完成这项操作。

⑥ 对可能要求一个不同参考带宽值的其他链路，可重复这个过程。

现在当运行仿真时，将依据参考带宽和链路带宽值，指派链路成本。

4）为每个路由器接口手工指定链路成本。链路成本是通过存在于所有路由器模型中的 OSPF 接口特定属性 **IP Routing Protocols... OSPF Parameters... Interface Information... <interface name>... Cost**（IP 路由协议... OSPF 参数... 接口信息... <接口名>... 成本）加以指定的。可为每个路由器接口手工编辑这个属性。典型情况下，仅当需要在少数接口上更新链路成本时，才可能希望实施这一步骤。也可检验这个属性，来验证成本已经采用上述的方法之一进行了正确的设置。

Timers	(...)
Hello Interval (seconds)	10
Router Dead Interval (seconds)	40
Interface Transmission Delay (seconds)	1.0
Retransmission Interval (seconds)	5.0

图 11.16　OSPF 接口定时器的配置属性

11.3.7　配置 OSPF 定时器

OPNET 支持如下四种 OSPF 定时器的配置。通过如图 11.16 所示的 OSPF 接口特定复合属性 **IP Routing Protocols... OSPF Parameters... Interface Information... <interface name>... Timers**（IP 路由协议 ... OSPF 参数 ... 接口信息 ... <接口名>... 定时器），可查看或配置这些 OSPF 定时器的值。通过指定如下子属性的值，可为一台路由器中每个个体接口配置 OSPF 定时器：

1）**Hello Interval**（**seconds**）［Hello 间隔（秒）］指定由这个接口产生的两个连续 Hello 消息之间的间隔。注意，对连接到同一网络的所有接口，这个值应该是相同的。对于所有接口类型，这个属性的默认值是 10s，另外情况是 MANET 接口，默认情况下，

它将这个值设置为 2s。

2）**Router Dead Interval**（**seconds**）［路由器死亡间隔（秒）］指定这样的时长，在这个时长之后，一台邻接路由器从之没有接收到 Hello 消息，则认为该路由器宕机。例如，如果在大于或等于 **Router Dead Interval**（**seconds**）值的时长之后，一台路由器没有看到来自一台邻接路由器的任何 Hello 消息，则那台邻接路由器就被认为是不活跃的。一般而言，这个值应该被设置为 Hello 间隔的某个倍数，且对连接到同一网络的所有接口都应该是相同的。这个属性的默认值是 40s（对 MANET 接口是 6s）。

3）**Interface Transmission Delay**（**seconds**）［接口传输延迟（秒）］指定在这个接口上传输一条 LSA 报文所花费时间量的一个估值。为了防止在洪泛过程中 LSA 的无限制循环，这个属性的值被用来老化一条 LSA（该 LSA 每次被一台路由器转发时）。默认情况下，这个属性被设置为 1s。

4）**Retransmission Interval**（**seconds**）［重传间隔（秒）］指定 LSA 重传之间的时间长度。默认情况下，这个属性被设置为 5s。

也可使用 **Protocols** 菜单配置 OSPF 定时器，如下所述：

1）选择希望以 OSPF 定时器值的相同集合配置的各链路。如果希望在网络中的所有连接接口上以相同方式配置 OSPF 定时器，则可忽略这一步骤。

2）选择 **Protocols → OSPF → Configure Interface Timers**（协议→OSPF→配置接口定时器）选项，打开如图 11.17 所示的 **Configure OSPF Timers** 窗口。

3）在相应的文本框中指定期望的定时器值。

4）通过分别选中 *All connected interfaces*（所有连接的接口）或 *Interfaces across selected links*（被选中链路间的接口）单选按钮，指定希望将所提供配置应用到所有连接的接口或被选中链路的接口。

5）单击 **OK** 按钮完成这项操作。

6）针对要求 OSPF 定时器不同值集合的其他链路，可重复这个过程。

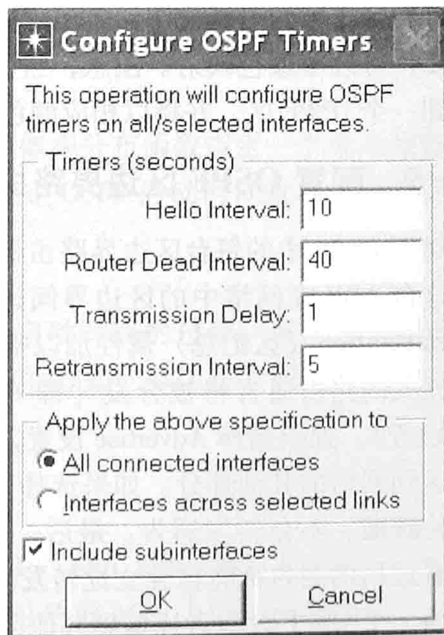


图 11.17　由 **Protocols** 菜单触发的 **Configure OSPF Timers** 窗口

11.3.8　配置 OSPF 区

通过实施如下步骤，可通过 **Protocols** 菜单配置 OSPF 区：

1）选择属于同一 OSPF 区的所有链路。

2）选择 **Protocols→OSPF→Configure Areas**（协议→OSPF→配置各区）选项，打开如图 11.18 所示的 **OSPF Area Configuration**（OSPF 区配置）窗口。

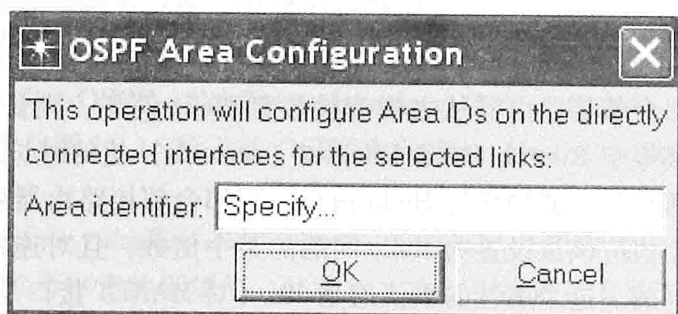


图 11.18　**OSPF Area Configuration**（OSPF 区配置）窗口

3）在 *Area identifier*（区识别符）文本框中指定期望的 OSPF 区号。注意仅有骨干 OSPF 区可被指派区号 0，而非骨干区可编号为 1、2、3 等（但不能是 0）。

4）单击 **OK** 按钮完成这项操作。

5）对于希望配置的每个 OSPF 区，重复这些步骤。但是，没有必要配置骨干区（区 0），原因是不指派到任何其他区的所有链路将默认地指派到区 0。

一旦配置了 OSPF 区，则通过在 **Project Editor** 中选择 **View→Visualize Protocols Configuration→OSPF Area Configuration**（查看→可视化协议配置→OSPF 区配置）菜单，可看到各区的可视化图示。出现的弹出窗口列出用于表示每个 OSPF 区的颜色。如果需要，可改变颜色映射。注意，当可视化时，仅有配置运行 OSPF 的那些接口才可被指派到一个 OSPF 区，并将以相应颜色进行标记。

11.3.9　配置 OSPF 区边界路由器

对于 OSPF 中的每台区边界路由器（ABR），可指定如下策略，确定路由通告如何通过这台 ABR 在网络中的区边界间进行发送。路由策略是通过路由器模型中的 **Area Summarization**（区汇总）属性加以指定的。对于一个给定区及其内部的地址范围，可指定相关的路由通告将被分发［即 Advertise（通告）设置］还是隐藏［即 Hide（隐藏）设置］。如果选择 Advertise 设置，那么到给定地址范围的所有路由都将使用单条通告发送到网络的其他部分。如果选择 Hide 设置，那么关注的地址范围将对网络其他部分保持隐藏，不会转发通告。最后，如果不配置 **Area Summarization**，那么默认情况下所有适用的通告都将被独立地转发。

当一个 OSPF 区包含连接到其他区的多台边界路由器，那么常见的做法是配置一些这样的 ABR 不要携带跨区流量。是这样做到这一点的，方法是为一个（或多个）ABR 指定策略，隐藏所有子网地址（对给定区是内部地址）。结果，这样一台路由器将不会将这些地址通告给该路由器所连接的其他区，且不从其他区接收目的地为这些子网地址的流量。

通过设置复合属性 **IP Routing Protocols... OSPF Parameters... Processes... <Process Tag>... Process Parameters... Area Summarization**（IP 路由协议...OSPF 参数...进程...<进程标签>...进程参数...区汇总），可为一台路由器配置路由通

告策略。这个属性由多行组成，每行指定单条路由器通告策略。每行由如下属性组成：

1）**Address**（地址）和 **Subnet Mask/Prefix Length**（子网掩码/前缀长度）为这个区汇总指定地址范围。

2）**Area ID**（区 ID）指定指派给这个地址范围的区的 ID。指定的区必须被连接到这台路由器。

3）**Status**（状态）指定路由器是向其他区通告还是隐藏这个地址区。这个属性仅接受两个值：Advertise（通告）和 Hide（隐藏）。

4）**Metric**（度量）指定与汇总地址相关联的度量。默认情况下，汇总度量被计算为被汇总的所有地址的最高度量。与汇总一起指定一个度量值的做法，覆盖默认计算。

11.3.10 配置 OSPF 启动时间

除了 11.3.3 节描述的通过属性 **Start Time**（启动时间）配置 OSPF 启动时间外，也可使用如下描述的 Protocols 菜单：

1）在希望指定 OSPF 启动时间的网络中选择支持 OSPF 的路由器。

2）选择 **Protocols→OSPF→Configure Start Time...**（协议→OSPF→配置启动时间...）选项，打开 **OSPF Start Time Configuration**（OSPF 启动时间配置）窗口。这个窗口几乎与图 11.8 所示配置 RIP 启动时间的窗口相同。

3）从分布的下拉列表中选择概率分布函数的名字，该函数被用来计算 OSPF 启动时间。

4）在 *Mean outcome*（均值输出）文本框中为概率分布函数指定一个输入参数。

5）通过选中相应的单选按钮，指定希望将指定 OSPF 启动时间配置应用到所有路由器还是在第一步中选中的路由器。

6）单击 **OK** 按钮应用配置并关闭窗口。

这个过程非常类似于 11.2.4 节所述配置 RIP 启动时间的过程。相比于直接改变相应属性，这是设置 OSPF 启动时间的一种比较方便的方式，原因是它可应用到所有路由器或路由器的一个子集。

11.3.11 OSPF 仿真效率模式

类似于 RIP，默认情况下，OPNET 将属性 **OSPF Sim Efficiency**（OSPF 仿真效率）设置为 Enabled，这意味着 OSPF 更新仅仿真运行时的前 260s [即更新间隔时长，可通过属性 **OSPF Stop Time (seconds)**（OSPF 停止时间（秒））进行配置] 在节点间进行交换。在此时间之后，不再产生 OSPF 更新。在 OSPF 不是研究对象的多数仿真中，对这些更新的抑制改善了仿真效率，其方法是降低必须由仿真内核处理的事件数。但是，如果关注于在时间进程中观察 OSPF 的行为，那么应该考虑禁止 OSPF 仿真效率模式，方法是执行如下步骤：

1）通过选择 **DES→Configure/Run DES**（DES→配置/运行 DES）菜单选项、按 Ctrl + R 键或另外一种方法即单击 *Running Man*（奔跑的人）图标，打开 **Configure/Run**

DES 对话框。

2）在 **Configure/Run DES** 对话框（即详细视图）中，展开属性 **Inputs... Global Attributes... Simulation Efficiency**（输入... 全局属性... 仿真效率）。

3）指定属性 **OSPF Sim Efficiency**（OSPF 仿真效率）和 **OSPF Stop Time**（**seconds**）［OSPF 停止时间（秒）］的值。现在可为 **OSPF Stop Time**（**seconds**）指定一个值（改变 OSPF 将停止的时间），或可完全地禁止仿真效率模式（通过将属性 **OSPF Sim Efficiency** 的值设置为 Disabled）。

4）单击 **Apply**（应用）按钮，在没有执行仿真的条件下保存配置改变，或单击 **Run**（运行）按钮，以新的配置执行仿真。

11.4 通用的路由统计量

OPNET 为评估路由协议提供全局统计量和节点统计量。具体而言，每个路由协议有一个独立的统计类，组合了与那个协议有关的所有统计量和节点统计量。表 11.1 提供了可用 RIP 统计量和 OSPF 统计量的一个汇总表。

表 11.1 RIP 和 OSPF 相关的全局及节点统计量汇总

类别	名称	描述
Global Statistics OSPF（全局统计量 OSPF）	*Network Convergence Activity*（网络收敛活动）	这个统计量记录一个方波，当在网络中任何地方都没有收敛活动时为 0，当网络中某个地方有收敛活动时为 1。收敛活动定义为路由表中的变化
	Network Convergence Duration（*sec*）［网络收敛时长（秒）］	这个统计量以秒为单位记录在整个网络间路由表收敛周期的时长
	Total OSPF Protocol Traffic Sent（*bits/sec*）［发送的 OSPF 总流量（比特/秒）］ *Total OSPF Protocol Traffic Sent*（*pkts/sec*）［发送的 OSPF 总流量（报文数/秒）］	这些统计量记录在网络中所有节点上所有连接的接口间发送的 OSPF 的流量总量。这些统计是以 bit/s 和报文数/s 为单位记录的
Global Statistics RIP（全局统计量 RIP）	*Network Convergence Activity*（网络收敛活动）	这个统计量记录一个方波，当在网络中任何地方都没有收敛活动时为 0，当网络中某个地方有收敛活动时为 1。收敛活动定义为路由表中的变化
	Network Convergence Duration（*sec*）［网络收敛时长（秒）］	这个统计量以秒为单位记录在整个网络间路由表收敛周期的时长
	Traffic Received（*bits/sec*）［接收的流量（比特/秒）］ *Traffic Sent*（*bits/sec*）［发送的流量（比特/秒）］	这些统计量记录在网络中所有节点上所有连接的接口间接收和发送的 RIP 的流量总量。这些统计是以比特/秒为单位记录的

（续）

类　　别	名　　称	描　　述
Global Statistics OSPF Advanced（全局统计量 OSPF 高级量）	*Database Description Traffic Sent*（*bits/sec*）［发送的数据库描述流量（比特/秒）］ *Database Description Traffic Sent*（*pkts/sec*）［发送的数据库描述流量（报文数/秒）］ *Hello Traffic Sent*（*bits/sec*）［发送的 Hello 流量（比特/秒）］ *Hello Traffic Sent*（*pkts/sec*）［发送的 Hello 流量（报文数/秒）］ *Link - State Acknowledgement*（*Multicast*）*Traffic Sent*（*bits/sec*）［发送的链路状态确认（组播）流量（比特/秒）］ *Link - State Acknowledgement*（*Multicast*）*Traffic Sent*（*pkts/sec*）［发送的链路状态确认（组播）流量（报文数/秒）］ *Link - State Acknowledgement*（*Unicast*）*Traffic Sent*（*bits/sec*）［发送的链路状态确认（单播）流量（比特/秒）］ *Link - State Acknowledgement*（*Unicast*）*Traffic Sent*（*pkts/sec*）［发送的链路状态确认（单播）流量（报文数/秒）］ *Link - State Request Traffic Sent*（*bits/sec*）［发送的链路状态请求流量（比特/秒）］ *Link - State Request Traffic Sent*（*pkts/sec*）［发送的链路状态请求流量（报文数/秒）］ *Link - StateUpdate*（*Multicast*）*Traffic Sent*（*bits/sec*）［发送的链路状态更新（组播）流量（比特/秒）］ *Link - StateUpdate*（*Multicast*）*Traffic Sent*（*pkts/sec*）［发送的链路状态更新（组播）流量（报文数/秒）］ *Link - StateUpdate*（*Unicast*）*Traffic Sent*（*bits/sec*）［发送的链路状态更新（单播）流量（比特/秒）］ *Link - StateUpdate*（*Unicast*）*Traffic Sent*（*pkts/sec*）［发送的链路状态更新（单播）流量（报文数/秒）］	这些统计量记录针对各种类型 OSPF 消息发送的流量总量。这些统计量是在网络中所有节点上的所有连接接口间记录的。为如下类型的 OSPF 消息收集这些统计量：数据库描述、Hello、链路状态 ACK（组播）、链路状态 ACK（单播）、链路状态请求、链路状态更新（组播）和链路状态更新（单播）。对于每种 OSPF 消息类型，这些统计量是以比特/秒和报文数/秒为单位记录的
	LSA Acknowledgement Rate（LSA 确认速率） *LSA Installation Rate*（LSA 安装速率） *LSA Origination Rate*（LSA 发起速率） *LSA Retransmission Rate*（LSA 重传速率）	这些统计量记录在一个 OSPF 区内所有接口处发生有关 LSA 各种事件的速率。具体而言，这些统计量是以如下 LSA 事件速率方式存在的： • *Origination rate*（发起速率）——发起 LSA 的速率 • *ACK rate*（ACK 速率）——发送 LSA 之 ACK 的速率 • *Retransmission rate*（重传速率）——当没有接收到 LSA 的 ACK 时，重传这些 LSA • *Installation rate*（安装速率）——当接收到一条新的更新时，将 LSA 安装到数据库中所发生的速率

（续）

类　别	名　称	描　述
Global Statistics OSPF Advanced（全局统计量 OSPF 高级量）	*Retransmission Traffic Sent*（*bits/sec*）[发送的重传流量（比特/秒）] *Retransmission Traffic Sent*（*pkts/sec*）[发送的重传流量（报文数/秒）]	这些统计量记录数据库同步报文的重传量（由所有 OSPF 接口发送的）。这些统计量是以 bit/s 和报文数/s 为单位记录的
Node Statistics OSPF（节点统计量 OSPF）	*Router Convergence Activity*（路由器收敛活动）	这些统计量记录一个方波，当在一台特定路由器上没有收敛活动时为 0，当在路由器处存在收敛活动时为 1
	Router Convergence Duration（*sec*）[路由器收敛时长（秒）]	这个统计量以 s 为单位记录在一台给定路由器处路由表收敛的时长
	Traffic Received（*bits/sec*）[接收到的流量（比特/秒）] *TrafficSent*（*bits/sec*）[发送的流量（比特/秒）]	这些统计量记录在一台给定路由器上运行的所有 OSPF 进程接收和发送的流量总量。这些统计量是以比特/秒为单位记录的
Node Statistics OSPF Process（节点统计量 OSPF 进程）	*Database Description Traffic Sent*（*bits/sec*）[发送的数据库描述流量（比特/秒）] *Database Description Traffic Sent*（*pkts/sec*）[发送的数据库描述流量（报文数/秒）] *Hello Traffic Sent*（*bits/sec*）[发送的 Hello 流量（比特/秒）] *Hello Traffic Sent*（*pkts/sec*）[发送的 Hello 流量（报文数/秒）] *Link－State Acknowledgement*（*Multicast*）*Traffic Sent*（*bits/sec*）[发送的链路状态确认（组播）流量（比特/秒）] *Link－State Acknowledgement*（*Multicast*）*Traffic Sent*（*pkts/sec*）[发送的链路状态确认（组播）流量（报文数/秒）] *Link－State Acknowledgement*（*Unicast*）*Traffic Sent*（*bits/sec*）[发送的链路状态确认（单播）流量（比特/秒）] *Link－State Acknowledgement*（*Unicast*）*Traffic Sent*（*pkts/sec*）[发送的链路状态确认（单播）流量（报文数/秒）] *Link－State Request Traffic Sent*（*bits/sec*）[发送的链路状态请求流量（比特/秒）] *Link－State Request Traffic Sent*（*pkts/sec*）[发送的链路状态请求流量（报文数/秒）] *Link－State Update*（*Multicast*）*Traffic Sent*（*bits/sec*）[发送的链路状态更新（组播）流量（比特/秒）] *Link－State Update*（*Multicast*）*Traffic Sent*（*pkts/sec*）[发送的链路状态更新（组播）流量（报文数/秒）] *Link－State Update*（*Unicast*）*Traffic Sent*（*bits/sec*）[发送的链路状态更新（单播）流量（比特/秒）] *Link－State Update*（*Unicast*）*Traffic Sent*（*pkts/sec*）[发送的链路状态更新（单播）流量（报文数/秒）]	这些统计量为各种类型 OSPF 消息记录发送的流量总量。为一台给定路由器上每个个体 OSPF 进程分别记录这些统计量。为如下类型的 OSPF 消息收集这些统计量：数据库描述、Hello、链路状态 ACK（组播）、链路状态 ACK（单播）、链路状态请求、链路状态更新（组播）和链路状态更新（单播）。对于每种 OSPF 消息类型，这些统计量是以 bit/s 和报文数/s 为单位记录的

（续）

类　别	名　称	描　述
Node Statistics OSPF Process（节点统计量OSPF进程）	*MANET Designated Router Status*（MANET指定的路由器状态）	这个统计量仅可用于MANET接口，它以方波的形式记录MANET指定路由器（MDR）的状态，可以为**MDR**状态（记录为2）；**Backup MDR**（备份**MDR**）（记录为1）；或**other MDR**（其他MDR）（记录为0）
	Traffic Sent（*bits/sec*）［发送的流量（比特/秒）］ *Traffic Sent*（*pkts/sec*）［发送的流量（报文数/秒）］	这些统计量记录一台给定路由器上由单个OSPF进程产生的流量总量。这些统计量是以bit/s和报文数/s为单位记录的
Node Statistics RIP（节点统计量RIP）	*Router Convergence Activity*（路由器收敛活动）	这个统计量记录一个方波，当在一台特定路由器处没有收敛活动时为0，当在路由器处有收敛活动时为1
	Router Convergence Duration（*sec*）［路由器收敛时长（秒）］	这个统计量以s为单位记录在一台给定路由器处路由表收敛话费的时长
	Traffic Received（*bits/sec*）［接收的流量（比特/秒）］ *Traffic Received*（*pkts/sec*）［接收的流量（报文数/秒）］ *Traffic Sent*（*bits/sec*）［发送的流量（比特/秒）］ *Traffic Sent*（*pkts/sec*）［发送的流量（报文数/秒）］	这些统计量记录运行在一台给定路由器上所有RIP进程接收和发送的流量总量。这些统计量是以bit/s和报文数/s为单位记录的

11.5　查看路由表

可输出各种与路由有关的报告，从而可在仿真结束之后进行查看。可输出由RIP、OSPF或部署在网络中的其他路由协议创建的路由表。也可输出OSPF LSDB、*IP Forwarding Table*（IP转发表）和其他网络相关的报告。可配置仿真在仿真的指定时间和/或在仿真结束时输出这些表的内容。另外，可配置仿真输出网络中特定的个体路由器或多个路由器的路由表。在本节，将仅讨论产生与RIP路由协议有关报告的步骤。同样的通用步骤适用于任何其他报告表的输出。

为配置个体路由器输出RIP相关的表，需要实施如下步骤：

1）选择希望输出其RIP路由信息的各路由器。

2）编辑所选中路由器中任何一台路由器的属性。

3）展开属性 **Reports**（报告），该属性包含配置路由器输出各种报告表的几个子属性。默认情况下，路由器被配置为根本不输出任何信息。

4）为在仿真结束时输出 *RIP Routing Table*（RIP 路由表），将属性 **RIP Routing Table** 的值设置为 Export at End of Simulation（在仿真结束时输出）。

5）另外，可能希望在仿真的其他时间输出 *RIP Routing Table*，在这种情形中需要展开属性 **RIP Routing Table**，并将其子属性 Status 设置为 Enabled。图 11.19 说明了这个过程。

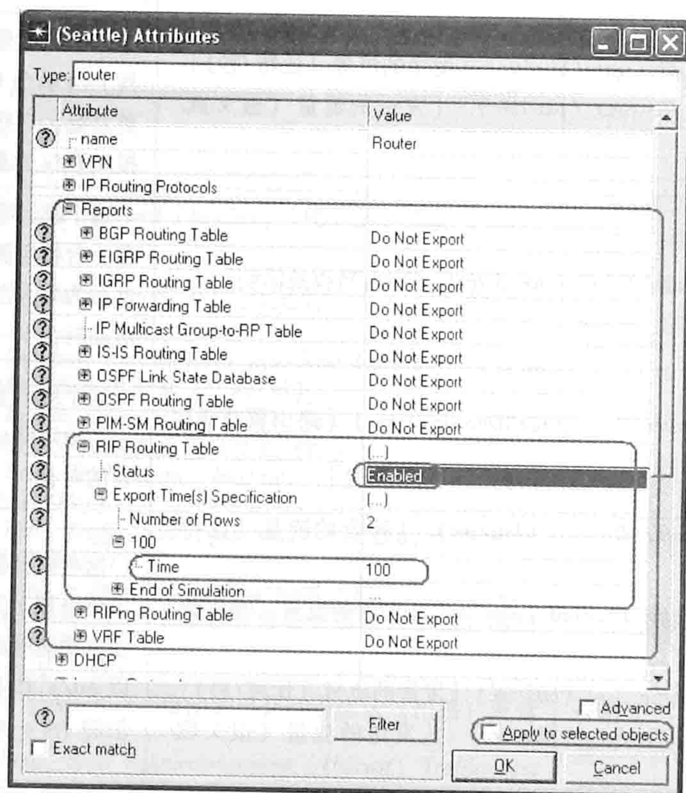


图 11.19 输出路由器报告

6）展开属性 **Export Time（s）Specification**（输出时间规格），并将属性 **Number of Rows** 的值设置为在仿真过程中希望输出 *RIP Routing Table* 的总次数。

7）展开每个新建的行，并将其属性 Time 设置为 *RIP Routing Table* 应该被输出的仿真时间。属性 Time 也接受值 End of Simulation（仿真结束），这将配置仿真在仿真结束处输出这台路由器的 *RIP Routing Table*。

8）选择检查框 *Apply to selected objects*（应用到被选中的对象），并在这个过程的第一步中选择的所有路由器上指定配置。

9）单击 **OK** 按钮，应用配置改变。注意，属性 **OSPF Routing Table**（OSPF 路由表）和 **OSPF Link State Database**（OSPF 链路状态数据库）分别负责输出 OSPF Routing Table 和 OSPF LSDB。

仅在仿真完成执行之后，才可查看所输出的路由器信息（如 *RIP Routing Table*），方法是实施如下步骤：

1）通过使用 **DES→Results→View Results**（DES→结果→查看结果）菜单选项，打开 **Results Browser** 窗口。

2）打开 *DES Run Tables* 选项卡，并展开 **Object Tables** 属性树，如图 11.20 所示。

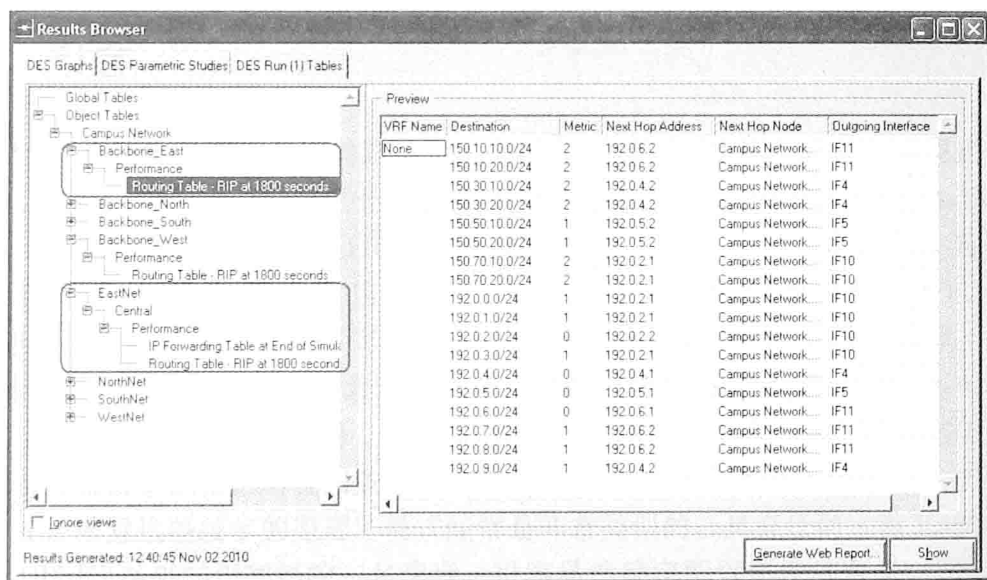


图 11.20　一个输出的 *RIP Routing Table*（*RIP* 路由表）预览

3）展开属性 **Performance**（性能），之后单击可用报告，显示在被选中节点上输出表的一个预览。如图 11.20 所示，*Preview* 平板显示 *Campus Network* 中节点 *Backbone _ East* 的 *RIP Routing Table*（这是输出的唯一路由器报告），而 *Campus Network* 的 *EastNet* 子网中节点 *Central* 包含 *IP Forwarding Table* 和 *RIP Routing Table*，这两个表在仿真结束时输出。

4）单击 **Show**（显示）按钮，在一个独立窗口中显示当前预览的报告。

5）也可单击 **Generate Web Report**（产生网页式报告）按钮，这使您可从这个场景中选择希望输出哪台路由器的报告，之后将所选中的信息在您计算上一个期望位置保存为一个 HTML 文件集合。

图 11.20 给出一个输出 *RIP Routing Table* 的例子，其中从标准 OPNET 项目 RIP 的 RIPv2 场景在时间 1800s 处输出 *Backbone _ East* 的表。将这个场景配置为，在仿真结束（运行 30min）时，输出网络内所有路由器上的 *RIP Routing Table*。如图 11.20 所示，所输出 *RIP Routing Table* 中的每行包含如下信息：

1）**Destination**（目的地）——由当前路由器可达的目的地网络（即 *Backbone _ East*）。

2）**Metric**（度量）——到达目的地网络的跳数。

3）**Next Hop Address**（下一跳地址）——在到目的地网络的路由上下一跳的 IP

地址。

4）**Next Hop Node**（下一跳节点）——在到目的地网络的路由上下一跳的一个逻辑名。

5）**Outgoing Interface**（外发接口）——接口的名字，在这个接口上一条报文将在其到最终目的地的路上被传输到达下一跳。



第 12 章　数据链路和物理层

12.1　引言

OPNET 支持大范围的数据链路层协议和技术，包括以太网、异步传递模式（Asynchronous Transfer Mode，ATM）、光纤分布式数据接口（Fiber Distributed Data Interface，FDDI）、帧中继（Frame Relay，FR）、令牌环（Token Ring，TR）、X.25 等。另外，OPNET 软件也包含仿真无线通信的各种模型，如无线局域网（WLAN）、WiMAX（802.16e）、移动自组织网络（Mobile AdHoc Network，MANET）、时分多址（TDMA，Time Dirision Multiple Access）、通用移动电信系统（Universal Mobile Telecommunications System，UMTS）等。典型情况下，每种这样的协议和技术都有指定一个期望配置的一个仿真属性集合。但是，所有相关参数和配置设置的一个完备描述超出了本书的范围。尽管如此，配置这些技术的原理和模式经常是非常类似的。具有任何一种上述协议最新知识和 OPNET 基础知识理解的一名读者，可容易地确定配置属性的含义，设计并配置一个网络的仿真模型，该网络利用了相应的数据链路层和无线通信技术。出于这个原因，本书并不详细描述所有可用的模型及其属性，相反，仅介绍在 OPNET 中部署数据链路层和无线通信技术背后的主要原理，并提供一些最普遍的协议和技术的一个概述。重要的是指出，某些无线协议族，如 WLAN、WiMAX 和 UMTS，要求独立的模块许可证。因此，除非存在无线模块许可证，否则 12.6 节和 12.7 节中描述的功能特征将是不存在的。欲了解所有支持的数据链路层和无线通信协议及技术的完备描述，参见 OPNET 的产品文档。

本章后面部分如下组织：在 12.2 节，描述部署和配置数据链路层节点及链路模型的通用原理。在 12.3 节，通过为链路上分析性能的链路配置属性和可用统计量，给出链路模型的一个概述。在 12.4 节和 12.5 节分别描述以太网和 TR 数据链路层技术。接下来的两节讨论无线网络：12.6 节描述 WLAN，而 12.7 节描述 MANET 技术。在 12.8 节说明如何定义节点移动性，在 12.9 节展示如何使用无线部署导引来创建新的无线网络。

12.2　采用数据链路层技术部署和配置仿真模型

为部署一种特定的数据链路层技术，要求在项目工作空间内放置支持那项技术的节点和链路模型。可依据模型的名字，识别由一个节点或链路模型所支持的数据链路协议。典型情况下，链路、端节点（如工作站和服务器）和集线器模型支持单一数据链路层技术，而交换机和层 3 设备可具有运行不同数据链路层技术的接口。例如，模型

ATM_E1、1000*Base_X_adv*、*FDDI*、*FR_link*、*TR*16 和 *sl_x25_int* 分别表示 ATM、以太网、FDDI、FR、TR 和 X. 25 链路。类似地，*atm_server_int*、*ethernet_wkstn*、*fddi_cache_server*、*fr_wkstn_adv* 和 *tr_unitx_wkstn* 是具有不同数据链路层借口的端节点模型的例子。集线器模型试验如下命名惯例：<*data link layer*><*number of ports*>_*hub*_<*suffix*>（<数据链路层><端口数>_*hub*_<扩展名>）。例如，具有 32 个端口的一个以太网集线器的高级模型命名为 *ethernet32_hub_adv*。路由器和交换机使用类似的命名惯例。但是，因为这样的设备支持多个接口，每个接口潜在地运行一种不同的数据链路层技术，模型名以一个下画线符号隔开每个不同的接口类型。例如，*eth2_fddi2_tr2_switch_adv* 是包含两个以太网接口、两个 FDDI 接口和两个令牌环接口的一台多用途（multipurpose）交换机的高级模型。

链路模型不包含配置数据链路层性质的属性。相反，这种配置属性位于节点模型中。典型情况下，仅有高级节点模型具有可用于配置数据链路层特征的属性，而常规节点使这些属性为隐藏的，并设置为默认值。例如，*tr_server_adv* 节点模型包含指定 TR 配置的复合属性 **Token Ring**（令牌环），而 *tr_server* 模型没有可用于配置的这个属性。另外，节点模型 *fr_server* 和 *fr_server_adv* 包含指定 FR 设置的复合属性 **Frame Relay**。一般而言，当研究数据链路层的性质时，使用高级节点模型是一种不错的想法。

正如您可想到的，当构建网络拓扑时，如果这些节点中的每个节点具有支持相同数据链路层技术的一个接口或一个端口，则可以一条链路连接两个节点。例如，不能直接连接 *ethernet_wkstn* 和 *tr_server*。为了在这两个节点之间建立一条通信信道，需要添加一个中间节点，如一台交换机或路由器。作为一个例子，对于中间节点，可使用诸如 *eth2_fddi2_tr2_switch* 或 *ethernet_tr_slip8_gtwy* 的节点模型，每个模型都包含至少一个以太网接口和一个令牌环接口。在中间节点添加到网络拓扑之后，可使用诸如 100*BaseT* 和 1000*BaseX* 的一个以太网链路模型将 *ethernet_wkstn* 连接到中间节点，并最后使用诸如 TR_adv 和 TR4 的 TR 链路模型将中间节点连接到 *tr_server*。

12.3 链路模型属性和特征

OPNET 包含几个链路类，包括双工、单工、总线和总线分支（bus tap）模型。典型情况下，模型的名字识别链路的类型和数据链路层技术（用来连接物理媒介）。例如，*eth_tap_adv* 代表一种高级以太网总线分支链路模型。类似地，*FR_T1_int* 代表以 T1 速率携带数据的一种中间 FR 双工链路模型。应该注意到，模型的名字不会表明链路是单工的还是双工的。但是，模型描述指定链路是单工的还是双工的。另外，单工链路被表示为末端有箭头的一条线，指明链路上流量断续流的方向，而双工模型被表示为没有任何箭头的一条线。

如图 12.1 所示，典型情况下，链路模型包含如下配置属性：

1）**transmitter**（发送器）、**transmitter a**（发送器 a）、**transmitter b**（发送器 b）识别附接到这条链路的一个节点或两个节点中的发送模块。仅在单工和总线分支模型中

存在 **transmitter** 属性，原因是数据流是单向的。仅在双工链路模型和总线分支模型中存在 **transmitter a** 和 **transmitter b** 属性。

2）**receiver**（接收器）、**receiver a**（接收器 a）、**receiver b**（接收器 b）识别附接到这条链路的一个或两个节点中的接收模块。**receiver** 属性仅存在于单工和总线分支模型中，而 **receiver a** 和 **receiver b** 属性仅存在于双工链路模型中。

3）**Propagation Speed**（传播速度）以米每秒为单位指定这种物理媒介的传播速度。这个属性仅在高级链路模型中是可见的。当属性 **delay**（延迟）被设置为 Distance Based（基于距离的）（这是默认值），这个属性的值和节点间的距离一起被用于确定链路上的传播延迟。

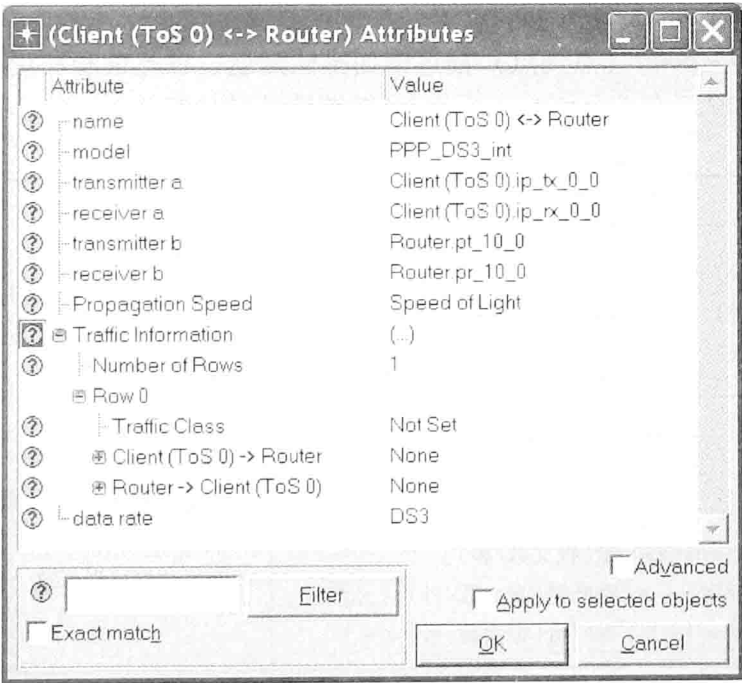


图 12.1　一个典型链路模型的配置属性

表 12.1　低层点到点链路统计量汇总

名　　字	描　　述
bit error rate（比特错误率）	这个统计量记录链路上的平均比特错误率
bit error rate per packet（每个报文的比特错误率）	这个统计量记录链路上每个报文的平均比特错误率
busy（忙） *busy* - >（忙 - >） *busy* < -（忙 < -）	这些统计量是作为布尔值 0 和 1 进行收集的，代表链路是空闲还是被占用（由于报文发送或接收）。箭头 < - 和 - > 指明在收集该统计量的链路上数据流的方向
packet loss ratio（报文丢失比率）	这个统计量也是作为布尔值 0 和 1 进行收集的。0 值表明在这条链路上的一次成功报文发送，而 1 值意味着一次报文发送是不成功的

4）**delay**（延迟）指定这条链路上的传播延迟。典型情况下，这个属性是隐藏的，

并被设置为默认值 Distance Based。为了查看和/或修改这个属性，需要选择高级检查框。若有必要，可显式地指定流量穿越这条链路时所经历的传播延迟值。

5）**Traffic Information**（流量信息）指定链路上的基线流量负载。欲了解有关这个属性配置的细节，请参见 6.6.3 节。

6）**data rate**（数据速率）以比特每秒指定链路上的传输速率。这个属性仅存在于高级链路模型中。某些链路模型，如以太网链路（如 1000BaseX、100BaseT 等），不包含这个属性。

OPNET 提供两种类型的链路统计量：

1）***low - level point - to - point***（低层点到点）描述物理信道低层特征的一组统计量。表 12.1 包含了 ***low - level point - to - point*** 链路统计量的汇总。

2）***point - to - point***（点到点）描述诸如排队延迟、吞吐量和利用率等链路特征的一组统计量。表 12.2 包含 ***point - to - point*** 链路统计量的汇总。

表 12.2　点到点链路统计量汇总

名　字	描　述
queuing delay（*sec*）[排队延迟（秒）] *queuing delay*（*sec*） - > [排队延迟（秒） - >] *queuing delay*（*sec*） < - [排队延迟（秒） < -]	这些统计量记录报文在链路队列中等待时间的即时测量。延迟是从报文进入链路队列的时间直到报文完成发送的时间进行测量的
throughput(*bits/sec*)[吞吐量(比特/秒)] *throughput*(*bits/sec*) - >[吞吐量(比特/秒) - >] *throughput*(*bits/sec*) < - [吞吐量(比特/秒) < -] *throughput*(*packets/sec*)[吞吐量(报文数/秒)] *throughput*(*packets/sec*) - >[吞吐量(报文数/秒) - >] *throughput*(*packets/sec*) < - [吞吐量(报文数/秒) < -]	这些统计量记录链路上的平均吞吐量。吞吐量定义为在链路上每秒成功接收或发送的比特数或报文数
utilization（利用率） *utilization* - >（利用率 - >） *utilization* < -（利用率 < -）	这些统计量记录链路利用率为 0% 和 100% 之间的一个值，其中 100% 意味着链路容量被全部消耗。链路利用率定义为在链路上携带的数据量和链路容量的比值

12.4　以太网

OPNET 实现了以太网协议的多数功能特征，例外是来去物理链路的数据传递的比特串行化。在 OPNET 中实现的以太网模型依据 IEEE 802.3、IEEE 802.3u 和 IEEE 802.3z 标准。具体而言，OPNET 支持如下以太网功能特征：发送请求的 FIFO 处理、基于链路长度计算传播延迟、载波侦听、冲突检测、重传尝试最大次数设置为 16 的二进制指数回退、延迟（deference）的帧间隙定时、冲突之后的垃圾序列发送、802.3 最小和最大帧尺寸、运行在半双工模式的 1000BaseX 以太网链路的帧突发，以及全双工和半双工数据传输。OPNET 也允许在所有通用网桥和交换机模型上配置基于端口的虚 LAN

(VLAN)，并提供对 Cisco 快速 EtherChannel（以太网信道）技术的支持。OPNET 仅允许以总线和集线（hub）方式部署以太网模型。欲了解 OPNET 中支持的所有以太网功能特征的一个完备列表，请参见产品文档。

OPNET 仅提供以太网的一些配置属性。这些属性仅在诸如 *eth2_fddi2_tr2_switch_adv*、*ethernet16_switch_adv* 和 *ethernet_wkstn_adv* 的高级节点模型中才是可见的。在复合属性 **Ethernet... Ethernet Parameters**（以太网... 以太网参数）下收集所有以太网配置属性。在支持多个以太网端口的节点模型中，单词 **Parameters**（参数）后跟在括号中的端口号。如图 12.2 所示，通过如下属性指定以太网配置：

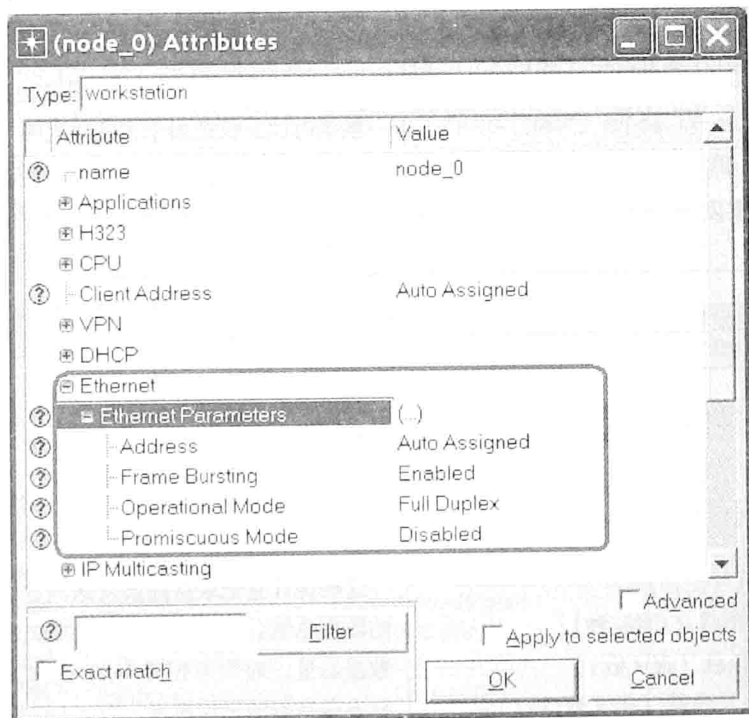


图 12.2　以太网配置属性

1）**Address**（地址）为这个以太网接口指定唯一的 MAC 地址。默认情况下，这个属性被设置为值 Auto Assigned（自动指派的），但如果需要，可显式地将这个属性设置为期望的 MAC 地址值。

2）**Frame Bursting**（帧突发）指定在这个接口上是否激活帧突发。帧突发仅适用于千兆以太网（IEEE 802.3z）。在所有其他场景中忽略这个属性值。当激活时，允许节点在这个接口上发送额外的帧，而不重启信道冲突过程。节点可继续发送，直到没有更多数据或直到突发定时器超时，这要看哪个事件先发生。

3）**Operational Mode**（工作模式）指定这个接口是工作在全双工模式（即可同时发送和接收帧）还是半双工模式（即在某个时间仅可实施数据帧的发送或接收）。这个属性仅接受两个值：Full Duplex（全双工）和 Half Duplex（半双工）。

4）**Promiscuous Mode**（混杂模式）指定在这个接口上是否激活混杂模式。这个属性仅接受两个值：Enabled（激活）和 Disabled（禁止）。当激活时，不管报文的目的地

MAC 地址为何，节点都要接收所有报文并将之转发给上层。典型情况下，在网桥和交换机节点的所有以太网接口上激活混杂模式。

OPNET 也为评估以太网性能包含一个节点统计量集合。这些统计量聚集在 **Ethernet**（以太网）类下。注意在没有节点带有以太网接口的仿真模型中，**Ethernet** 节点统计量类是隐藏的。表 12.3 包含 **Ethernet** 节点统计量的汇总。

表 12.3 以太网节点统计量汇总

名 字	描 述
Burst Duration（*sec*）[突发时长（秒）]	这个统计量记录以太网接口处于突发模式的时间量。这个统计量是针对每次突发计算的，即从突发的第一个帧开始发送的时间直到突发结束其发送的最后一帧的时间。仅适用于千兆以太网
Burst ON/OFF（突发开/关）	这个统计量将以太网接口的突发状态记录为布尔值的一个序列：当没有突发发送时为 0，当存在突发发送时为 1。仅适用于千兆以太网
Size（*packets*）[突发尺寸（报文数）]	这个统计量记录每次突发发送过程中的报文数。仅适用于千兆以太网
Collision Count（冲突计数）	这个统计量记录从这个以太网接口发送的数据所经历的冲突总数
Delay（*sec*）[延迟（秒）]	这个统计量记录到达这个以太网接口的帧所经历的延迟
Load（*bits*）[负载（比特）] *Load*（*bits/sec*）[负载（比特/秒）] *Load*（*packets*）[负载（报文数）] *Load*（*packets/sec*）[负载（报文数/秒）]	这些统计量记录从高层发送到这个接口上以太网层的数据总量。当单位是 bit 或报文数时，记录发送的数据总量，而当单位是 bit/s 或报文数/s 时，记录每秒发送数据的平均速率
Traffic Received（*bits*）[接收到的流量（比特）] *Traffic Received*（*bits/sec*）[接收到的流量（比特/秒）] *Traffic Received*（*packets*）[接收到的流量（报文数）] *Traffic Received*（*packets/sec*）[接收到的流量（报文数/秒）]	这些统计量记录在这条链路上从这个接口接收到并由以太网层发送到高层的数据量。当单位是 bit 或报文数时，记录接收的数据总量，而当单位是 bit/s 或报文数/s 时，记录每秒接收数据的平均速率
Transmission Attempts（发送尝试次数）	这个统计量记录在报文的目的地处成功接收报文之前由这个工作站进行的发送尝试次数

12.5 令牌环

OPNET 包括对 TR MAC 联网协议的支持。所提供的模型，在以环状或集线状拓扑组织的网络中，工作在 4Mbit/s 或 16Mbit/s 的发送速率下。OPNET 的 TR 实现基于 IEEE

802.5 标准。但是，在 OPNET 并没有实现所有的 TR 功能特征，原因是模型主要仅针对仿真和性能估计而设计。具体而言，OPNET 没有显式地对环初始化和恢复处理进行建模。网络管理（NMT）和 MAC 之间的接口也没有实现。没有针对 NMT 的错误检测和错误报告的支持。为了改善仿真性能，TR 模型实现令牌加速功能，在没有数据发送时的时间段过程中，该功能阻塞令牌传递。当数据发送重启时，令牌被注回 TR LAN。

TR 模型实现了 MAC 和逻辑链路控制（LLC）子层之间的一个接口以及 MAC 和物理（PHY）层之间的一个接口。另外，它也显式地对如下功能特征进行了建模，如 LLC PDU 的可定义优先级等级、一个 24 比特可确保的最小延迟、站延迟和传播延迟的效应、重新堆栈操作以及令牌持有定时器（THT）。

OPNET 仅提供 TR 的一些配置属性。这些属性仅存在于高级节点模型中，如 *eth2_fddi2_tr2_switch_adv*、*tr16_bridge_adv* 和 *tr_wkstn_adv*。所有 TR 配置属性都被收集在复合属性 **Token Ring... Token Ring Parameters**（令牌环... 令牌环参数）下。在具有多个 TR 端口的节点模型中，单词 **Parameters** 后跟带括号的端口数。如图 12.3 所示，可通过如下属性指定 TR 配置：

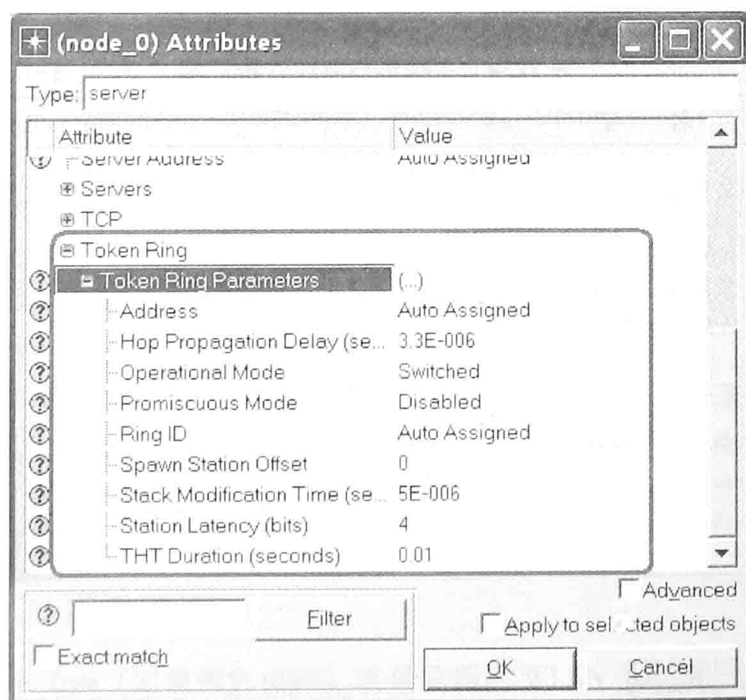


图 12.3 令牌环配置属性

1）**Address**（地址）是在当前 TR 接口上指定唯一的 TR MAC 地址。默认情况下，这个属性被设置为 Auto Assigned，但如有必要，可显式地指定期望的 MAC 地址值。

2）**Hop Propagation Delay（seconds）**［跳传播延迟（秒）］指定一帧从一个 TR 接口传播到下一个接口所花费的时间。

3）**Operation Mode**（工作模式）指定这个接口是工作于 Switched（交换式的）（即可同时发送和接收帧）还是 Shared（共享的）（即在某个时间仅可实施数据的发送

或接收）模式。这个属性仅接受两个值：Switched 和 Shared。

4）**Promiscuous Mode**（混杂模式）指定在这个接口上是否激活混杂模式。

5）**Ring ID**（环 ID）指定这个 TR 接口所属环的标志（identity）。

6）**Spawn Station Offset**（衍生站偏移）指定用来选择 TR 接口的值，创建令牌并将之注入环。

7）**Stack Modification Time**（**seconds**）［栈修改时间（秒）］指定一个 TR MAC 修改其栈所需的时间。

8）**Station Latency**（**bits**）［站延迟（比特）］以比特为单位，指定一条报文在其到目的地的路径上由它所访问的每个 TR MAC 所引入的延迟。依据连接的接口速度（例如 4Mbit/s 或 16Mbit/s），比特值被转换为时间。

9）**THT Duration**（**seconds**）［THT 时长（秒）］指定一个 TR 接口在将令牌释放给下一个 TR 站之前可使用该令牌的最大时间量。这个值经常被称作令牌持有时间。

OPNET 也包含评估 TR 性能的一个节点统计量集合。这些统计量聚集在 **Token Ring** 类下。注意，在没有带有 TR 接口的节点的仿真场景中，**Token Ring** 节点统计类是隐藏的。表 12.4 包含 **Token Ring** 节点统计类的一个汇总。

表 12.4　令牌环节点统计量汇总

名　字	描　述
Delay（*sec*）［延迟（秒）］	这个统计量记录到达这个令牌环接口上的帧所经历的延迟
Load（*bits*）［负载（比特）］ *Load*（*bits/sec*）［负载（比特/秒）］ *Load*（*packets*）［负载（报文）］ *Load*（*packets/sec*）［负载（报文数/秒）］	这些统计量记录从高层发送到这个接口上令牌环层的数据总量
Traffic Received（*bits*）［接收到的流量（比特）］ *Traffic Received*（*bits/sec*）［接收到的流量（比特/秒）］ *Traffic Received*（*packets*）［接收到的流量（报文数）］ *Traffic Received*（*packets/sec*）［接收到的流量（报文数/秒）］	这些统计量记录在这个接口从令牌环层发送到高层的数据总量

12.6　无线局域网

OPNET 软件包括 WLAN 模型套件，但需要一个无线模块许可证，才能运行采用这个套件的仿真。在本节描述的有关 WLAN 的所有功能特征以及下一节有关 MANET 的功能特征，仅当有这个许可证时才是可用的，如果没有这个许可证，则它们是不可见的。

WLAN 模型套件的实现基于 IEEE 802.11、802.11a、802.11b、802.11g 和 802.11e 标准，它支持如下特征：

1）带有指数回退机制的载波侦听多路访问和冲突避免（CSMA/CA）分布式协同功能（DCF）访问方案。

2）点协同功能（PCF）访问方案。

3）混合协同功能（HFC），具有基于优先级冲突访问之四个访问类（AC）的支持增强的分布式信道访问（EDCA），传输机会（Transmission Opportunity）（TXOP），帧突发和访问点（AP）的EDCA参数设置分配。

4）用于可靠数据传输的基于阈值的请求发送/清除发送（RTS/CTS）交换。

5）可选的数据帧分片。

6）AP功能。即使所有WLAN节点都有AP能力，但仅有WLAN网桥、交换机或路由器才可将基站子系统（BSS）连接到外部网络。

7）漫游能力，其中一个WLAN节点可扫描其他AP，且如果当前AP的信号强度低于一个可接受水平时，则节点可切换到具有较强信号的另一个AP。

8）MAC层确认：正常ACK、阻塞ACK和无ACK。

9）802.11b、802.11g和/或802.11e节点之间的互操作能力。

10）物理层技术：跳频扩频（frequency - hopping spread spectrum，FHSS）、红外（IR）、直接序列扩频（DSSS）、正交频分复用（OFDM）和扩展速率PHY - OFDM。

11）差分移相键控（DPSK）、二相移相键控（BPSK）、正交移相键控（QPSK）、正交幅度调制（QAM）和互补码键控（CCK）调制技术。

12）1Mbit/s、2Mbit/s、5.5Mbit/s、6Mbit/s、9Mbit/s、11Mbit/s、12Mbit/s、18Mbit/s、24Mbit/s、36Mbit/s、48Mbit/s和54Mbit/s的数据速率。

13）地貌建模。默认情况下，OPNET使用自由空间传播模型。但是，如果有地貌建模模块（TMM）的许可证，那么可使用其他模型，如Longley - Rice和地貌集成粗糙地球模型（TIREM）。可通过**Topology→Terrain**（拓扑→地貌）菜单选项管理地貌建模，仅当存在TMM许可证时这个菜单才出现。

但是，其他功能特征，如背景流量、跳频、功率节省模式、认证和安全、节点失效和恢复、WLAN节点中的可变传输速率、某些802.11e功能特征以及其他一些功能特征，在OPNET的WLAN模型套件中都是不支持的。欲了解WLAN模型的功能特征和限制的一个完全列表，请参见OPNET产品文档。

12.6.1 WLAN配置属性

Object Palette Tree（对象调色板树）将最常用的WLAN节点模型保存在*wireless_lan*、*wireless_lan_adv*和*wlan*对象调色板中。存在几种不同的WLAN节点：

1）无线工作站和服务器模型，如*wlan_server*和*wlan_wkstn_adv*。这些节点模型包含TCP/IP参考模型栈的所有层。

2）无线终端站模型，如*wlan_station_adv*。这些节点模型仅包含链路层和物理层WLAN功能以及直接产生和接收流量的能力，用到显式报文产生（见6.5节）。这些模型是针对仿真研究设计的，它们的焦点主要为WLAN MAC和物理层的评估。

3）无线路由器和网桥（如*wlan_fddi2_tr2_router*、*wlan_eth_bridge*），包含无线接口和有线接口（即以太网、TR、SLIP等）。

WLAN 节点配置设置被组合在复合属性 **Wireless LAN**（无线局域网）下，它由如图 12.4 所示的两个子属性组成：

1）**Wireless LAN MAC Address**（无线局域网 MAC 地址）指定一个唯一的 WLAN MAC 地址。默认情况下，这个属性被设置为 Auto Assigned，但如有必要，可指定一个期望的 WLAN MAC 地址值。

2）**Wireless LAN Parameters**（无线局域网参数）指定这个 WLAN 接口的物理特征。

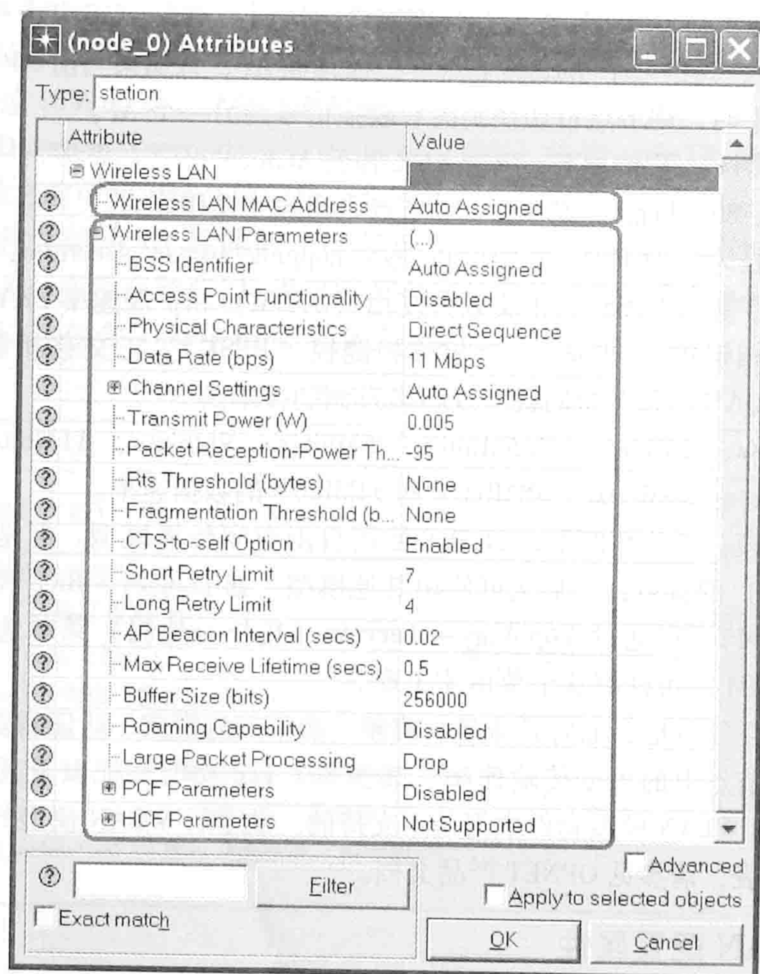


图 12.4 WLAN 配置属性

Wireless LAN Parameters 也是一个复合属性，由如下子属性组成：

1）**BSS Identifier**（BSS 识别符）指定这个 WLAN 节点所属的 BSS 的身份。默认情况下，这个属性被设置为 Auto Assigned，其中在当前子网内的所有节点属于同一个 BSS。如果在网络中的任意一个节点中，将这个属性设置为一个非默认值，那么也需要在网络中所有其他节点中显式地指定 BSS id。在漫游节点的情形中，这个属性值指定初始 BSS id。当节点移动时，它可被指派到一个不同的 BSS。

2）**Access Point Functionality**（访问点功能）指定在这个节点上是激活还是禁止

AP 功能。为指定 AP 功能，OPNET 文档指出如下规则：

① 每个 BSS 至多有一个 AP。一个 BSS 可没有 AP。

② 如果一个 BSS 被连接到其他网络或它部署 PCF，那么要求有一个节点激活 AP 功能。

③ 在一个网桥、交换机或路由器上的一个 WLAN MAC 接口必须激活 AP 功能。这个规则的一种例外情况是，在一台路由器中的一个 WLAN 接口，其中该路由器是一个 WLAN 骨干的组成部分。

④ 在一个节点中激活 AP 功能，会产生额外的仿真处理额外负担。当在仿真中不显式要求时，建议禁止这个选项。

3）**Physical Characteristics**（物理特征）识别在这个接口上部署的物理层技术。这个属性接受如下值：Frequency Hopping（跳频）（当前不支持）、Direct Sequence（直接序列）、Infra Red（红外）、OFDM（802.11a）和 Extended Rate PHY（802.11g）［扩展速率 PHY（802.11g）］。

4）**Data Rate**（**bps**）［数据速率（bit/s）］指定在这个 WLAN 接口上的传输速率。可接受数据速率值集合取决于所部署的物理层技术（是通过属性 **Physical Characteristics** 配置的）。

5）**Channel Settings**（信道设置）是一个复合属性，指定这个 WLAN 接口上由发送器和接收器使用的频带。欲了解附加细节，请参见属性描述。

6）**Transmit Power**（**W**）［发送功率（瓦）］属性确定这个 WLAN 节点的发送距离。它以瓦为单位指定这个节点的信号发送功率。

7）**Packet Reception - Power Threshold**（**dBm**）［报文接收功率阈值（dBm）］指定最小信号功率值，这使到达的报文可为接收器接受。以小于这个属性值的一个功率值的任意报文都被认为是噪声，并可能导致干扰。仅有大于或等于阈值的一个功率的报文才为这个 WLAN 接口接受和处理。

8）**Rts Threshold**（**bytes**）［Rts 阈值（字节）］指定最小帧尺寸，包括 WLAN MAC 首部（28B），该尺寸触发 RTS/CTS 交换。在被发送之前，其尺寸超过这个属性值的任何报文都必须等待一次成功的 RTS/CTS 交换完成。这个属性接受值 None（无），这指明在这个节点上没有使用 RTS/CTS 帧交换。

9）**Fragmentation Threshold**（**bytes**）［分片阈值（字节）］指定在不分片条件下可被发送的一帧的最大尺寸。尺寸大于这个属性值的每个帧，都被分片为尺寸为 **Fragmentation Threshold**（**bytes**）的数据块，例外情况是可能的最后一片。但是，除非属性 **Large Packet Processing**（大型报文处理）设置为 Fragment，否则大于 MAC 服务数据单元（MSDU）尺寸（为 2304B）的所有数据报文都将被丢弃。

10）**CTS - to - self Option**（CTS 到自身选项）确定在这个 WLAN 接口上是否激活 CTS - to - self 保护机制。仅当 WLAN MAC 工作在 802.11g 模式时，这个属性才是适用的。如果这个属性设置为 Disabled，那么执行一次常规的 RTS/CTS 帧交换，例外情况是不管这个属性的值为何，802.11g MAC 广播发送都使用 CTS - to - self 机制。

11）**Short Reply Limit**（短应答限制）为尺寸小于或等于 **RTS Threshold** 值的各帧，指定发送尝试的最大次数。如果在 **Short Reply Limit** 尝试次数之后，帧发送失败，则接口放弃尝试发送帧，并丢弃该帧。

12）**Long Reply Limit**（长应答限制）为尺寸大于 **RTS Threshold** 值的各帧，指定发送尝试的最大次数。

13）**AP Beacon Interval（seconds）**［AP 信标间隔（秒）］指定目标信标发送时间（TBTT）值，它确定 AP 的两次信标发送之间的时间长度。

14）**Max Receive Lifetime（seconds）**［最大接收寿命（秒）］指定从第一个分片到达，到放弃并丢弃一个部分接收的帧之前，节点必须等待所有分片到达的时长。

15）**Buffer Size（bits）**［缓冲尺寸（比特）］为存储来自上层的数据，指定最大缓冲尺寸。当缓冲满时，直到在缓冲中有可用的更多空间之前，来自上层的所有到达报文都将被丢弃。

16）**Roaming Capability**（漫游能力）指定这个节点是否激活漫游能力。在一个自组织 BSS 和激活 PCF 的一个 BSS 中，不支持漫游。

17）**Large Packet Processing**（大型报文处理）指定如何处理大于最大 MSDU 尺寸的报文。这个属性接受两个值：

① Drop（丢弃）为默认值，指明丢弃尺寸大于最大 MSDU 的所有报文。

② Fragment（分片）指明该节点将尝试对大型报文分片。仅当属性 **Large Packet Processing**（大型报文处理）设置为 Fragment 且在这个 WLAN 接口上激活分片［如属性 **Fragmentation Threshold（bytes）** 设置为非 None 的其他值］时，该节点才对一个大型报文分片。

18）**PCF Parameters**（PCF 参数）是一个复合属性，指定 PCF 模式的配置。默认情况下，这个属性设置为 Disabled。

19）**HCF Parameters**（HCF 参数）是一个复合属性，指定在 IEEE 802. 11e 标准中定义的 HCF 模式之 WLAN QoS 机制的配置。默认情况下，这个属性设置为 Not Supported（不支持）。

OPNET 也包含如图 12. 5 所示的四个全局 WLAN 属性。仅当仿真包含 WLAN 对象模型时，这些属性才是可用的；否则，它们是隐藏的且不可用于配置。在 **Configure/Run DES** 窗口的 *Global attributes*（全局属性）选项卡中可访问的属性如下：

1）**Closure Method（non - TMM）**［闭包方法（非 - TMM）］指定将使用哪种方法确定两个 WLAN 节点之间的通信是否可能［当不存在 TMM（要求一个独立的软件许可证）时］。这个属性接受如下两个值：

① No Occlusion（无阻塞）指定每对无线节点应该能够通信，即仿真中任何两个节点之间的传输路径都是无阻塞的。这是 WLAN 中的默认设置。

② Earth Line - of - Sight（地球视距）采用一个椭圆地球模型的视距，计算任意两个节点之间的闭包。当使用这种设置时，确保正确地配置节点高度，以避免因为地球曲面导致阻塞出现两节点之间通信失败的情况。OPNET 建议当使用 Earth Line - of - Sight

闭包方法时要配置所有 WLAN 节点具有一个非零的高度值。

2）**WLAN AP Connectivity Check Interval**（**seconds**）[WLAN AP 连通性检查间隔（秒）] 指定 WLAN 节点验证其与 AP 的连通性并扫描替代 AP（如有必要）的两次连续尝试之间的时间长度。仅当禁止信标效率模式 [即属性 **WLAN Beacon Transmission Count**（WLAN 信标传输计数）设置为 **Periodic**（周期性的）] 且在 WLAN 节点中激活漫游时，这个属性才是适用的。

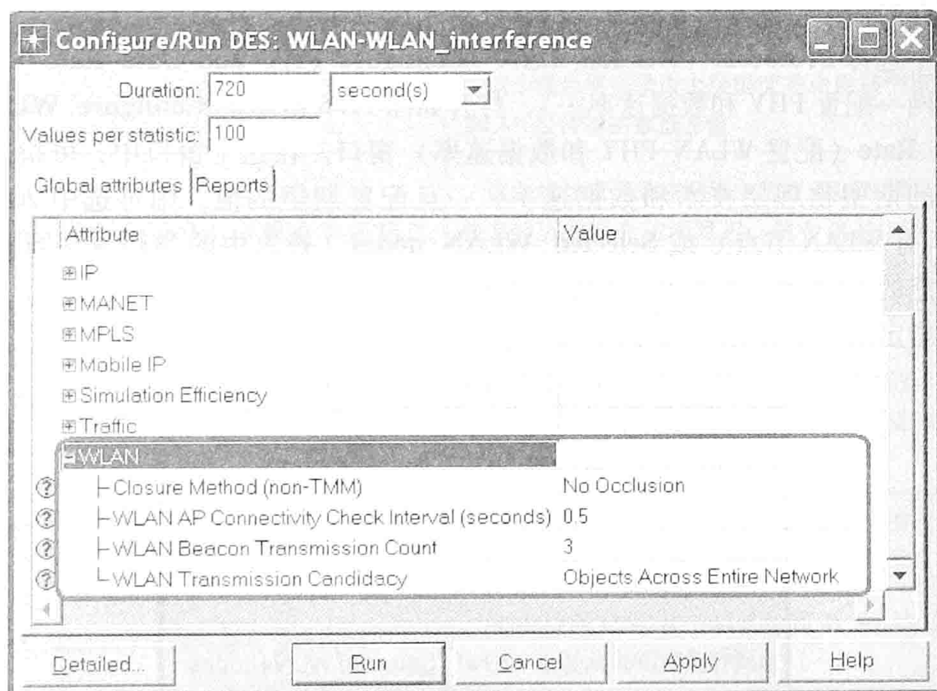


图 12.5　指定 WLAN 设置的全局配置属性

3）**WLAN Beacon Transmission Count**（WLAN 信标传输计数）指定是否激活信标效率模式。这个属性接受如下值：

① 一个正整数值指明信标效率模式是激活的（即将产生信标的数量）。这个值指定在不连续的信标传输之前，AP 将产生的信标数量。对于这种设置，漫游将是可能的。节点检查来自附近 AP 的信号强度，在没有信标的显式广播条件下，确定何时需要切换到另一个 AP。

② Periodic（周期性的）值指明禁止信标效率模式，且依据节点属性 **AP Beacon Interval**（**seconds**）[AP 信标间隔（秒）] 的配置值，AP 将周期性地产生一个信标。

欲了解有关这个属性的附加信息，请参见属性文档。

4）**WLAN Transmission Candidacy**（WLAN 传输候选权）属性允许激活仿真效率，这将消除属于不同子网的节点间的任何通信和/或干扰。实际上，仅有在同一子网内的节点才能够相互通信，并影响同一子网内节点的传输（即引入干扰）。这个属性接受如下两个值：

① Objects in Same Subnet（同一子网内的对象）指明激活仿真效率，且各节点仅

能与同一子网内的其他节点通信并影响它们的通信。在一个子网外的一个节点对那个子网内的通信没有影响。

② Objects Across Entire Network（在整个网络间的对象）指明禁止这种仿真效率，且和正常的情况一样执行仿真。在这种情形中，来自一个节点的传输可能导致其他子网中节点处的干扰（如果那些节点处在这个节点的干扰范围内）。这是这个属性的默认值。

另外，OPNET 包含一个 **Protocols** 菜单选项，用于配置仿真中所有或仅有被选中 WLAN 节点的物理特征和数据速率。可实施这个配置步骤，方法是从 **Project Editor** 的下拉菜单中选择 **Protocols→Wireless LAN→Configure PHY and Data Rate…**（协议→无线局域网→配置 PHY 和数据速率…），打开如图 12.6 所示的 **Configure WLAN PHY and Data Rate**（配置 WLAN PHY 和数据速率）窗口。在这个窗口中，可指定要在网络中配置的期望物理层技术和数据速率。一旦配置期望的值，则可选中 **All WLAN nodes**（所有 WLAN 节点）或 **Selected WLAN nodes**（被选中的 WLAN 节点）单选按钮，指明您的配置设置将应用到网络中的所有或仅有被选中的 WLAN 节点。单击 **OK** 按钮，在指定的 WLAN 节点中更新相应属性的值。

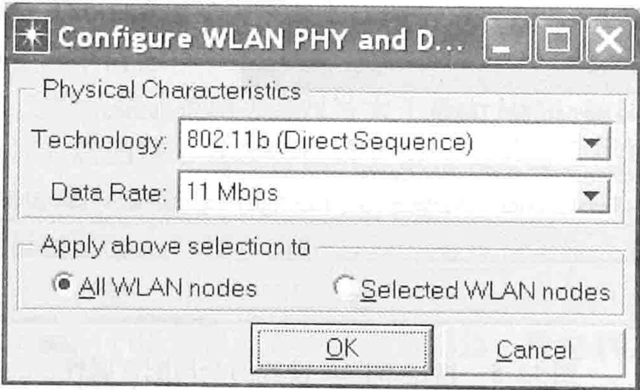


图 12.6　**Configure WLAN PHY and Data Rate**（配置 WLAN PHY 和数据速率）窗口

12.6.2　WLAN 统计量

OPNET 为评估 WLAN 性能提供全局和节点统计量。全局和节点统计量包含两个统计类，每个都可用于分析 WLAN 性能：

1）**Wireless LAN**（无线局域网）包含有关整体 WLAN 性能的统计量。在表 12.5 和表 12.6 分别给出全局统计量和节点统计量的汇总。

表 12.5　全局 WLAN 统计量汇总

名　字	描　述
Data Dropped（*Buffer Overflow*）（*bits/sec*）［丢弃的报文（缓冲上溢）（比特/秒）］	这个统计量记录由于高层缓冲上溢［通过节点属性 **Buffer Size**（**bits**）控制缓冲尺寸的］或由于数据报文超出最大 MSDU 尺寸，从高层接收并由网络中的所有 WLAN 节点丢弃的数据总量

（续）

名　　字	描　　述
Data Dropped（*Retry Threshold Exceeded*）（*bits/sec*）［丢弃的数据（超过重传阈值）］	这个统计量记录由于重复的失败重传（即超过相应的短重试或长重试阈值），从高层接收并由网络中的所有 WLAN 节点丢弃的数据总量
Delay（*sec*）［延迟（秒）］	这个统计量记录网络中所有 WLAN 接口成功接收的所有报文所经历的端到端延迟
Load（*bits/sec*）［负载（比特/秒）］	这个统计量记录由上层提交并由网络中所有节点上 WLAN 层传输的数据总量
Media Access Delay（*sec*）［媒介访问延迟（秒）］	这个统计量记录在网络中所有 WLAN 接口上所传输报文经历的媒介访问延迟。这个值是如下计算得到的，从报文被插入到传输队列的时间，直到报文被首次发送到物理层的时间，这两个时间形成的时间区间
Network Load（*bits/sec*）［网络负载（比特/秒）］	这个统计量是在每个 BSS 基础上计算的。它代表来自于高层，被整个 WLAN BSS 接收、接受和为传输而排队的数据量
Retransmission Attempts（*packets*）［重传尝试（报文数）］	这个统计量记录网络中由所有 WLAN 节点所做的重传尝试总数
Throughput（*bits/sec*）［吞吐量（比特/秒）］	这个统计量记录在网络的所有 WLAN 节点中从 WLAN 层转发到高层的数据量

表 12.6　节点 WLAN 统计量汇总

名　　字	描　　述
AP Connectivity（AP 连通性）	这个统计项记录当前节点连接到的 AP 的 id。如果节点没有连接到任何 AP，那么记录值 -1。当在扫描模式中正在搜索一个令人满意的 AP 时，认为该节点没有连接
Backoff Slots（*slots*）［回退时间槽（槽数）］	这个统计量记录站点为竞争媒介，在能够发送帧之前，需要回退的时间槽数
Control Traffic Rcvd（*bits/sec*）［接收到的控制流量（比特/秒）］ *Control Traffic Rcvd*（*packets/sec*）［接收到的控制流量（报文数/秒）］ *Control Traffic Sent*（*bits/sec*）［发送的控制流量（比特/秒）］ *Control Traffic Sent*（*packets/sec*）［发送的控制流量（报文数/秒）］	这些统计量记录在这个 WLAN 接口上接收或发送的控制流量总量。控制流量包括这样的帧，如 RTS、CTS、ACK、CF - End、CF - End + CF - ACK、Block - ACK 请求和 Block - ACK

（续）

名　字	描　述
Data Dropped (*Buffer Overflow*) (*bits/sec*) [丢弃的数据（缓冲上溢）（比特/秒）] *Data Dropped* (*Buffer Overflow*) (*packets/sec*) [丢弃的数据（缓冲上溢）（报文数/秒）]	这些统计量记录由于上层缓冲被完全占用或由于上层报文尺寸超过最大MSDU尺寸，由这个WLAN接口丢弃的数据流量总量
Data Dropped (*Retry Threshold Exceeded*) (*bits/sec*) [丢弃的数据（超过重传阈值）（比特/秒）] *Data Dropped* (*Retry Threshold Exceeded*) (*packets/sec*) [丢弃的数据（超过重传阈值）（报文数/秒）]	这些统计量记录由于重复的失败重发（即超过相应的短重试或长重试阈值），由这个WLAN接口从高层接收之后由该接口丢弃的数据总量
Data Traffic Rcvd (*bits/sec*) [接收的数据流量（比特/秒）] *Data Traffic Rcvd* (*packets/sec*) [接收的数据流量（报文数/秒）]	这些统计量记录在这个WLAN接口上从物理层成功接收的WLAN数据流量。当以比特/秒为单位报告这些统计量时，物理和MAC首部尺寸被包括在所接收流量总量的计算中。这些统计量不管目的地址为何记录WLAN接口上接收的所有数据
Data Traffic Sent (*bits/sec*) [发送的数据流量（比特/秒）] *Data Traffic Sent* (*packets/sec*) [发送的数据流量（报文数/秒）]	这些统计量记录由这个WLAN接口上发送到物理层的数据量。当以比特/秒为单位报告这些统计量时，物理和MAC首部尺寸被包括在所发送流量总量的计算中
Delay (*sec*) [延迟（秒）]	这个统计量记录在这个WLAN接口上从物理层成功接收并转发到上层的所有报文所经历的端到端延迟
Load (*bits/sec*) [负载（比特/秒）] *Load* (*packets/sec*) [负载（报文数/秒）]	这些统计量记录由上层提交到这个WLAN进行传输的数据总量
Management Traffic Dropped (*bits/sec*) [丢弃的管理流量（比特/秒）] *Management Traffic Dropped* (*packets/sec*) [丢弃的管理流量（报文数/秒）]	这些统计量记录由WLAN MAC丢弃的非数据流量总量。这些统计量记录所发送管理帧的丢失（作为802.11e Block - ACK方案执行的结果）。因此，仅当当前WLAN接口支持802.11e协议并激活Block - ACK方案时，才可记录这些统计量
Management Traffic Rcvd (*bits/sec*) [接收到的管理流量（比特/秒）] *Management Traffic Rcvd* (*packets/sec*) [接收到的管理流量（报文数/秒）]	这些统计量记录这个接口上从物理层接收的管理流量（即信标和Block - ACK帧）总量。这些统计量记录管理帧的成功到达，即使它们的目的地不是这个接口。当以比特/秒为单位报告这些统计量时，在接收到的流量总量计算中包括物理首部和MAC首部尺寸

（续）

名　字	描　述
Management Traffic Sent（*bits/sec*）［发送的管理流量（比特/秒）］ *Management Traffic Sent*（*packets/sec*）［发送的管理流量（报文数/秒）］	这些统计量记录这个接口发送到物理媒介的管理流量（即信标和 Block - ACK 帧）总量。当以比特/秒为单位报告这些统计量时，在发送的流量总量计算中包括物理首部和 MAC 首部尺寸
Media Access Delay（*sec*）［媒介访问延迟（秒）］	这个统计量记录提交到这个 WLAN 接口进行传输的报文所经历的媒介访问延迟。这个值计算为从报文被插入传输队列的时间到报文被首次发送到物理层的时间间隔
Queue Size（*packets*）［排队尺寸（报文数）］	这个统计量记录在这个 WLAN 接口发送队列中被缓冲的报文总数
Retransmission Attempts（*packets*）［重传尝试（报文数）］	这个统计量记录在这个 WLAN 接口上重传尝试总次数（即直到报文被成功发送或由于达到短重试或长重试阈值限制而被丢弃）
Throughput（*bits/sec*）［吞吐量（比特/秒）］	这个统计量记录成功从物理媒介接收并从这个 WLAN 接口转发到高层的数据总量

2）**WLAN（Per HCF Access Category）**［WLAN（每个 HCF 访问类）］包含 WLAN 统计量，是为每个 HCF AC 独立地收集的。这些统计量与常规 WLAN 统计量相同，因此本书略去其描述。

OPNET 产品文档提供了创建和部署 WLAN 网络基本步骤的绝好概述。另外，称为 WLAN 的一个范例项目，包含几个场景，说明在各种设置下 WLAN 网络的配置和评估。

12.7　MANET

MANET 是以自组织方式排列的一组 WLAN 节点，它们相互通信，形成一个网络。在一些情形中，MANET 节点可通过一个网关连接到一个有线（wired）网络。OPNET 支持如下 MANET 路由协议：开放最短路径优先版本 3（OSPFv3）、自组织应需距离向量（AODV）、动态源路由（DSR）、地理路由协议（GRP）和临时排序的路由算法（TORA）。但是，OPNET 在同一节点上不支持多个 MANET 路由协议，且对路由协议间路由再发布提供非常有限的支持。欲了解所支持 MANET 路由协议功能特征的一个完全列表，请参见 OPNET 的产品文档。

Object Palette Tree 在 MANET 对象调色板下保持最常用的支持 MANET 的节点模型。有三种不同类型的支持 MANET 的节点：

1）WLAN 服务器和工作站，如 *wlan_ server* 和 *wlan_ wkstn*。这些节点模型支持标准应用，如 HTTP、FTP 和远程登录，并可运行任意 MANET 路由协议。

2）MANET 站，如 *manet_ station_ adv*。这些模型显式地产生网络流量［通过复合属性 **MANET Traffic Generation Parameters**（MANET 流量产生参数）］，可以是流量连续流（stream）的一个源、一个中间节点或一个目的地，并可运行任何 MANET 路由协议。

3）WLAN 路由器和 MANET 网关节点模型（如 *manet_ gtwy_ wlan_ ethernet_ slip4*）主要作为从 MANET 到一个 IP 网络的 AP。

可通过节点属性和全局属性（通过 **Configure/Run DES** 对话框访问）指定 MANET 配置。MANET 节点配置通过如图 12.7 所示的如下复合属性进行指定：

1）**AD－HOC Routing Parameters**（自组织路由参数）指定在这个节点上部署的 MANET 路由协议和那个协议的配置设置。取决于部署的是哪个协议，每个 MANET 路由协议包含另一个复合子属性，如 **AODV Parameters**（AODV 参数）、**DSR Parameters**（DSR 参数）和 **GRP Parameters**（GRP 参数）。由于篇幅限制，略去路由协议配置属性的描述。请参见 OPNET 产品文档，了解支持的 MANET 路由协议功能特征和可用配置参数的完整描述。

2）**Wireless LAN**（无线局域网）指定 WLAN 接口的物理特征。请参见 12.6 节了解这些属性的描述。

3）**MANET Traffic Generation Parameters**（MANET 流量产生参数）指定由这个节点产生的流量特征。请参见 6.6.3 节了解这些属性的描述。

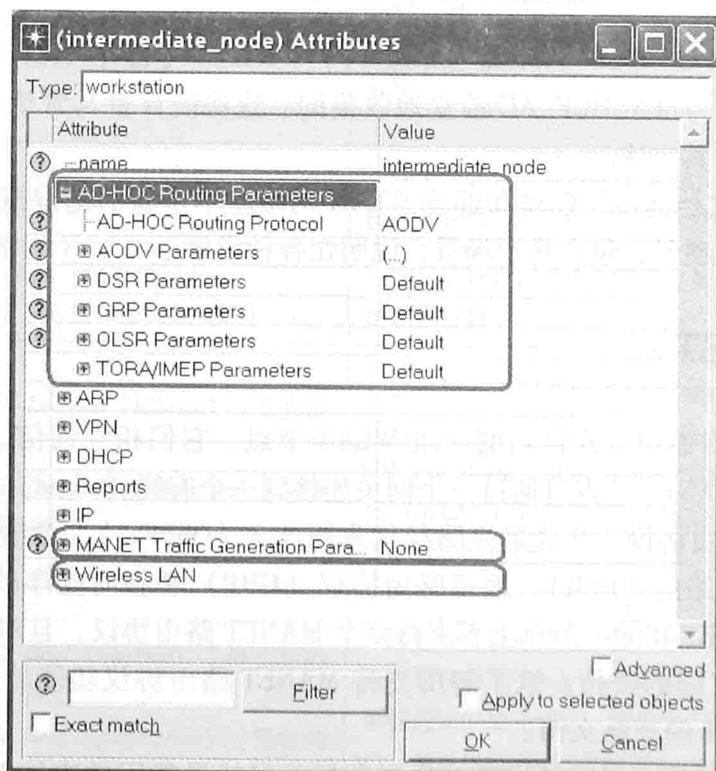


图 12.7　全部 MANET 配置属性

如图 12.8 所示，用于指定 MANET 配置的全局属性，主要处理作为一个整体应用到仿真场景的路由协议规格的各个方面。这些配置属性的含义或者是自解释的或协议特定的属性，在本书中略去对它们的描述。

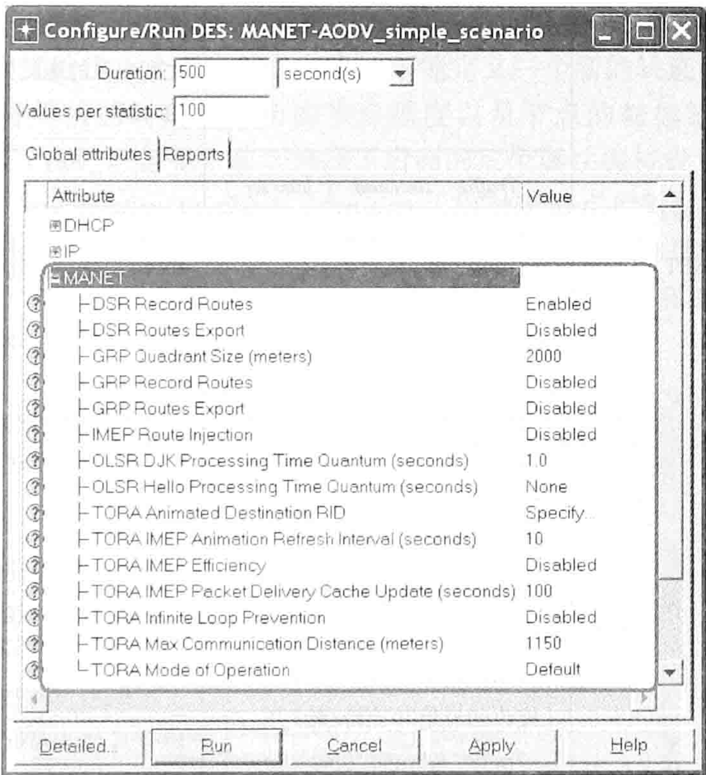


图 12.8　指定 MANET 设置的全局配置属性

另外，OPNET 也提供显示或隐藏 DSR 和 GRP 路由的一个选项。通过 **Protocols→MANET→DSR→Display DSR Routes…/Hide DSR Routes**（协议→MANET→DSR→显示 DSR 路由…/隐藏 DSR 路由）和 **Protocols→MANET→GRP→Display GRP Routes…/Hide GRP Routes**（协议→MANET→GRP→显示 GRP 路由…/隐藏 GRP 路由）选项，可仿真这项功能。

OPNET 为分析 MANET 和 MANET 路由协议提供全局统计量和节点统计量。这里略去评估 MANET 路由协议的可用统计量的描述。但是在表 12.7 中有 **MANET** 统计量的一个汇总。全局统计量和节点统计量包括称为 **MANET** 的一个类，包含评估 MANET 性能的统计量。仅当一个仿真场景包含至少一个 MANET 节点模型时，MANET 统计量才是可用于收集的。

OPNET 产品文档为部署 MANET 仿真模型，提供基本步骤的一个绝佳概述。另外，称为 MANET 的一个范例项目，包含几个场景，说明了各种设置下 MANET 网络的配置和评估。

表 12.7　MANET 统计量汇总

类　型	名　称	描　述
Global Statistics MANET（全局统计量 MANET）	*Delay*（*secs*）[延迟（秒）]	这个统计量记录任意 MANET 报文穿越网络所经历的端到端延迟。端到端延迟计算为在源处产生 MANET 报文到在目的地销毁该报文之间所消逝的时间
	Traffic Received（*bits/sec*）[接收的流量（比特/秒）] *Traffic Received*（*packets/sec*）[接收的流量（报文数/秒）]	这些统计量记录在整个网络中所有 MANET 流量目的地所接收的流量总量
	Traffic Sent（*bits/sec*）[发送的流量（比特/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	这些统计量记录在整个网络中所有 MANET 流量源所发送的流量总量
Node Statistics MANET（节点统计量 MANET）	*Delay*（*secs*）[延迟（秒）]	这个统计量记录到达这个节点的 MANET 报文所经历的端到端延迟
	Traffic Received（*bits/sec*）[接收的流量（比特/秒）] *Traffic Received*（*packets/sec*）[接收的流量（报文数/秒）]	这些统计量记录这个节点从网络中所有其他 MANET 节点接收的流量总量
	Traffic Sent（*bits/sec*）[发送的流量（比特/秒）] *Traffic Sent*（*packets/sec*）[发送的流量（报文数/秒）]	这些统计量记录在由这个节点发送到网络中所有其他 MANET 节点的流量总量

12.8　指定节点移动性

任意 WLAN 节点可以是固定的或移动的。固定节点在整个仿真期间保持静止，而移动节点随着仿真进行而到处移动。可通过属性 **trajectory**（轨迹）指定在仿真过程中节点的运动，该轨迹包含预设运动轨迹的一个列表。如果需要，可通过 **Topology→Define Trajectory…**（拓扑→定义轨迹…）菜单选项指定自己的节点运动轨迹，或通过 **Topology→Clear Trajectory Assignment…**（拓扑→清除轨迹指派…）选项清除所部署的轨迹。

12.8.1　定义一个节点轨迹

为定义一个新的轨迹，实施如下步骤：

1）从 **Project Editor** 下拉菜单中选择 **Topology→Define Trajectory**…选项。

2）在出现的 **Define Trajectory**（定义轨迹）窗口（见图 12.9）中指定轨迹名字。注意，在没有为轨迹给出一个名字的条件下，不能定义一个新的轨迹。也可指定节点在开始其运动之前将等待的时间量、初始节点高度以及节点的初始滚动（roll）、坡度（pitch）和偏航角（yaw）。但是，典型情况下后面的三个属性保持设置为其默认值。选择称为 *Coordinates are relative to object's position*（相对于节点位置的坐标）的检查框，也是一个不错的想法。选择这个选项，将确保节点轨迹中的起点将是相对于节点初始位置的。结果，每个节点（被指派相同的轨迹）将遵循相同的地理和时间特征的一条路径，但开始于工作空间中一个不同的（即其自己的）位置。

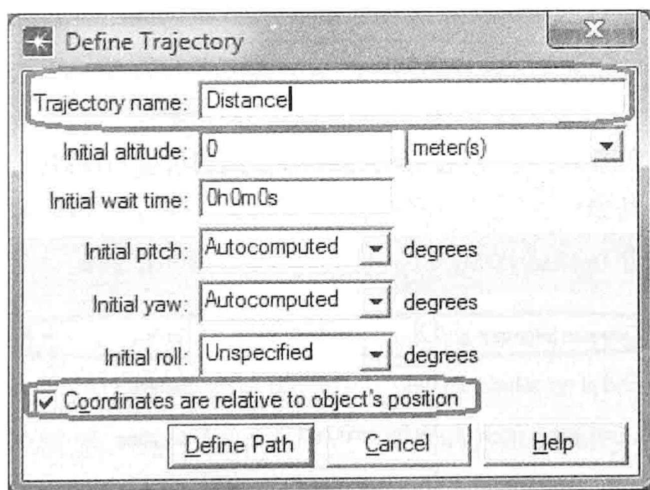


图 12.9　**Define Trajectory** 窗口

3）单击 **Define Path**（定义路径）按钮指定轨迹路径的第一段。这打开一个 **Trajectory Status**（轨迹状态）窗口（见图 12.10），并设置一条蓝线，旁边有单词 start（开始）（当鼠标指向该线时）。可为这个轨迹分段修改节点运动的速度或时长，并定义通过 **Trajectory Status** 窗口测量距离所用的单位。

4）在项目工作空间中希望轨迹开始处的一个位置单击。

5）将轨迹拖动通过工作空间，直到到达这个轨迹段应该结束的位置，并单击那个位置。在工作空间中将出现为这个轨迹段定义路径的一个红色箭头。另外，也将打开 **Segment Information**（分段信息）窗口（见图 12.11）。

6）在 **Segment Information**（分段信息）窗口中，当节点穿越这个轨迹段时可指定节点速度，以及当节点到达这个分段结束时节点的高度、滚动、坡度（pitch）和偏航角（yaw）。也可指定节点在这个位置应该保持静态的时间。

7）在指定当前轨迹段之后，可实施如下动作之一：

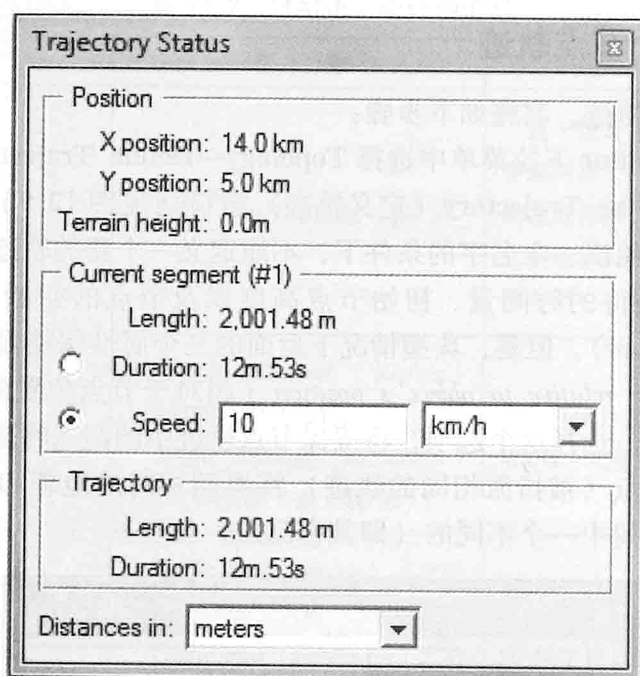


图 12.10 **Trajectory Status**（轨迹状态）窗口

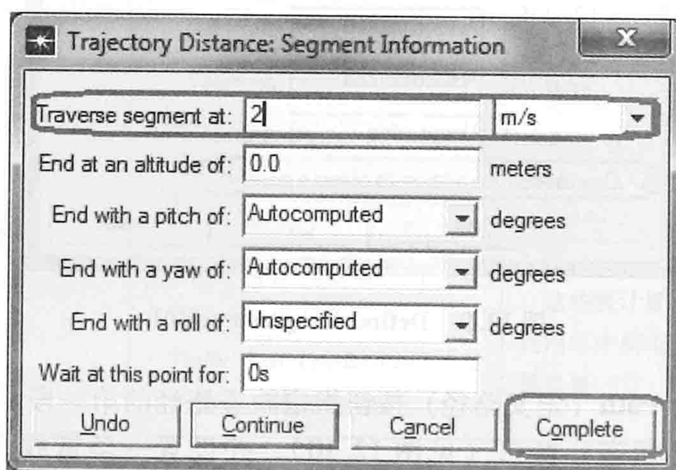


图 12.11 **Segment Information**（分段信息）窗口

① 单击 **Undo** 按钮，重新定义这个轨迹段。除了初始轨迹段外，可在任何轨迹段上实施这项操作。

② 单击 **Continue** 按钮，定义这个轨迹中的下一段。

③ 单击 **Cancel** 按钮，在不保持的条件下终止这个轨迹的定义。

④ 单击 **Complete** 按钮，完成并保持这个轨迹定义。所创建的轨迹定义被保存为默认模型目录中扩展名为 .try 的一个文本文件。

一旦定义了新的轨迹，可将之指派给一个期望的移动节点，方法是将节点的 **trajectory** 属性设置为新定义轨迹的名字。将需要在所有已定义轨迹的列表（当尝试设置这

个属性的值时出现）中寻找期望轨迹的名字。同样要记住属性 **trajectory** 仅存在于移动节点中。如果期望这样做的话，同一轨迹可指派给多个节点。在这样一种情形中，如果轨迹是采用相对于节点位置的坐标定义的，那么这个轨迹所指派到的每个节点将从其自己初始的位置开始，并遵循所定义轨迹相同的几何特征和时间特征的一条路径。另外，如果轨迹是采用绝对坐标定义的，那么所有这样的节点将遵循与其初始起点位置无关的同样路径。

12.8.2　配置一个移动性概要

与采用指定的坐标（绝对的或相对的）定义一个轨迹的做法不同，可选择配置一个节点依据一个移动性概要在仿真过程中随机移动。菜单选项的 **Topology→Random Mobility…**（拓扑→随机移动性…）设置，使您可部署或清除移动性概要并重用使用移动性概要创建的一个轨迹。

具体而言，存在如下三个菜单选项：

1）**Set Mobility Profile...**（设置移动性概要 ...）打开 **Configure Mobility Profile on Selected Nodes...**（在选中的节点上配置移动性概要）窗口，这使您可指定希望在被选中节点上部署哪个移动性概要。注意，在执行这项操作之前，必须选择一个或多个移动节点。默认情况下，可以如下三种移动性概要之一配置选中的节点：Default Random Waypoint（默认随机航路点）、Random Waypoint（Record Trajectory）[随机航路点（记录轨迹）] 或 Static（静态）。

2）**Clear Mobility Profile...**（清除移动性概要 ...）打开一个 **Clear Mobility Profile...** 窗口，这使您在执行这项操作之前从网络中的所有节点或仅从那些被选中的节点清除一个移动性概要。通过在相应节点上将属性 **trajectory** 设置为值 NONE，OPNET 清除一个移动性概要。

3）**Set Trajectory Created from Random Mobility...**（设置从随机移动性产生的轨迹 ...）使您可重用在以前仿真执行过程中记录的一条轨迹。当希望使用在仿真的以前运行过程中使用一个移动性概要产生的相同轨迹而重新运行一个仿真时，这个选项是有用的。注意，为了使这项操作正常工作，首先需要部署一个移动性概要 [记录产生的轨迹，即概要属性 **Record Trajectory**（记录轨迹）设置为 Enabled]，并至少执行一次仿真，从而轨迹得以记录下来。

除了三个默认的移动性概要，即 Default Random Waypoint、Random Waypoint（Record Trajectory）和 Static 之外，也可定义自己的移动性概要。当以 **Set Mobility Profile...**（设置移动性概要 ...）菜单选项选择默认概要之一时，一个 *Mobility Config*（移动性配置）节点自动地添加到工作空间（如果该节点还不是仿真场景组成部分的话）。另外，可显式地将一个 *Mobility Config* 节点添加到仿真场景。无论在哪种情形，之后都可定义一个新的移动性概要，方法是在 *Mobility Config* 节点的复合属性 **Random Mobility Profile** 中添加一新行，并将其属性（见图 12.12）设置为期望的值。这些属性的含义是自解释的，因此这里略去不讲。可用于指派到无线节点的随机移动性概要的数量，

对应于 *Mobility Config* 节点中概要表项的数量。除了三个默认的概要外，添加的概要也出现在 **Configure Mobility Profile on Selected Nodes...**（在选中节点上配置移动性概要...）窗口中。

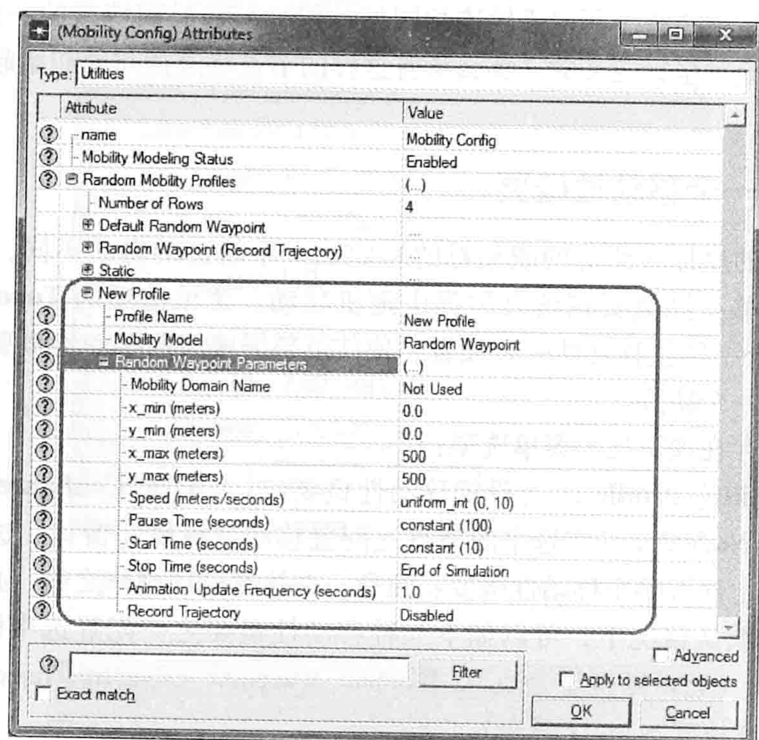


图 12.12　定义一个新的移动性概要的配置属性

12.9　使用无线部署导引

OPNET 包含一个无线部署导引，这极大地简化了创建和部署无线网络场景的任务。WLAN 和 MANET 均可采用这个导引进行创建。为启动无线部署导引，选择 **Topology→Deploy Wireless Network...**（拓扑→部署无线网络...）菜单选项，打开如图 12.13 所示的窗口。部署过程由通过一系列配置屏幕指定无线网络细节组成。每个窗口包含其目的的一个简短描述。当正使用导引部署无线网络时，将有几个可用的按钮（当适用时）：

1）**Quit**（退出）按钮——在不保存所做改变的情况下，终止无线网络部署。

2）**Back/Next**（上一步/下一步）按钮——相应地移到前或下一配置屏幕。

3）**Help**（帮助）按钮——打开一个帮助窗口，提供在当前配置屏幕中可用选项的一个简短描述。

4）**Finish**（完成）按钮——完成配置过程，并在项目工作空间中部署所指定的无线网络。

现在，回顾一下使用无线部署导引的过程：

1）通过选择 **Topology → Deploy Wireless Network...**（拓扑→部署无线网络...）选项，启动无线部署导引。

2）单击 **Continue** 按钮前进到下一屏幕，这使您可指定从一个文件中载入无线网络规格还是使用导引创建一个崭新的网络定义。

3）如果打开一个以前保存的配置文件，那么所有配置屏幕将采用从那个文件中载入的值进行预设置。否则，所有配置屏幕将包含默认值。无论从一个文件载入网络规格还是使用导引创建一个新的网络规格，为了部署无线网络，将仍然需要走过所有的配置屏幕。

4）第一个配置屏幕使您可在项目工作空间内指定无线网络的位置。

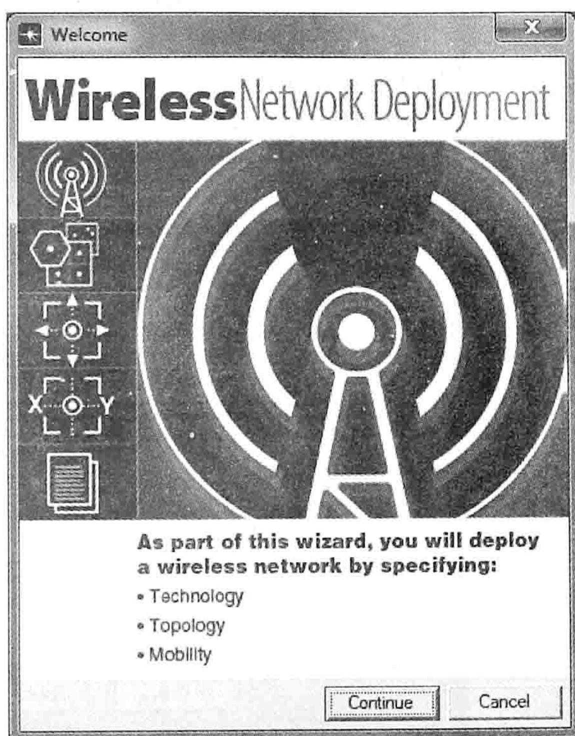


图 12.13　无线部署导引的 **Welcome**（欢迎）窗口

5）如图 12.14 所示的下一配置屏幕，可指定在网络中采用的无线技术。可通过一个下拉菜单，包含诸如 WLAN（Ad－hoc）［WLAN（自组织）］、WLAN（Infrastructure）［WLAN（基础设施）］、WiMAX 和 TDMA（Distributed）［TDMA（分布式的）］的各种值选择无线技术。应该使用 WLAN（Ad－hoc）创建一个 MANET 网络，使用 WLAN（Infrastructure）创建带有基站基础设施的一个 WLAN。当从下拉列表中选择无线技术时，配置属性集合将发生改变。如果期望的话，则可将这些属性设置为不同值。在所部署无线网络中的所有节点将相应属性设置为在这个配置屏幕中指定的值。

6）下一个配置屏幕使您可在网络内指定节点的放置（见图 12.15）。可指定诸如区的尺寸（在其中将部署无线网络）和节点放置等信息。这是无线网络配置的一个至关重要的方面，原因是网络性能取决于通信距离（range）。例如，将节点相互放置得太远，会导致不良的性能，原因是接收到的信息将会太弱。另外，将节点相互放置得太近，会导致每个节点都能够与所有其他节点通信，导致大量干扰，或使您不能观察到路由协议行为，原因是网络中的所有路由都是一跳远的。请注意外观和可用配置值取决于在前面各屏幕中的选择。

7）如图 12.16 所示的下一配置屏幕，使您可指定在无线网络中要使用的节点模型的类型和数量。

8）在此之后，可配置在无线网络中的各节点在仿真过程中将如何移动。可配置节点依据移动概要之一而随机移动，或为节点指定实际的运动轨迹。如图 12.17 所示，通过改变移动配置表中相应单元（cell）的值，也可指定在无线网络中有多少节点将移

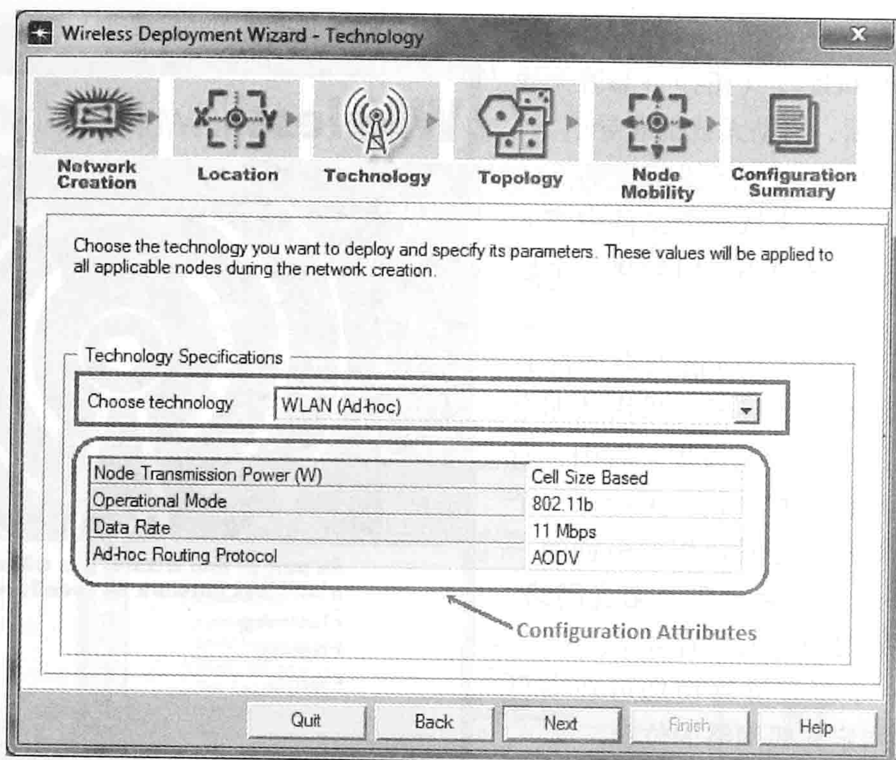


图 12.14　无线部署导引的 **Technology** 配置屏幕

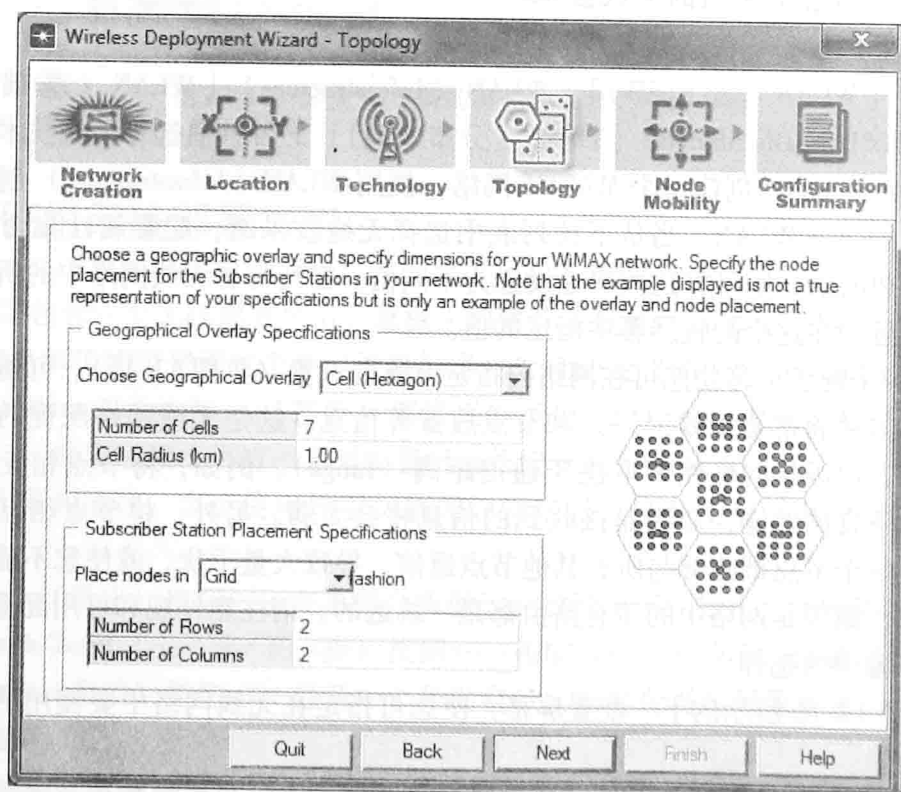


图 12.15　指定节点配置的无线部署导引的 **Topology** 配置屏幕

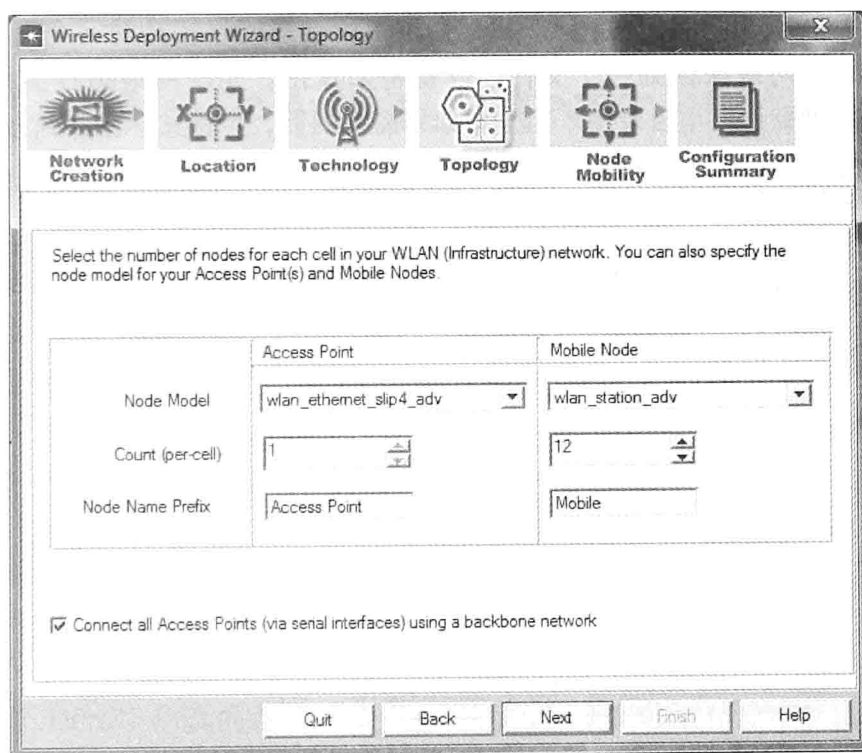


图 12.16　为指定节点的类型和数量无线部署导引的 **Topology** 配置屏幕

动、它们的速度和高度以及它们的运动区范围。

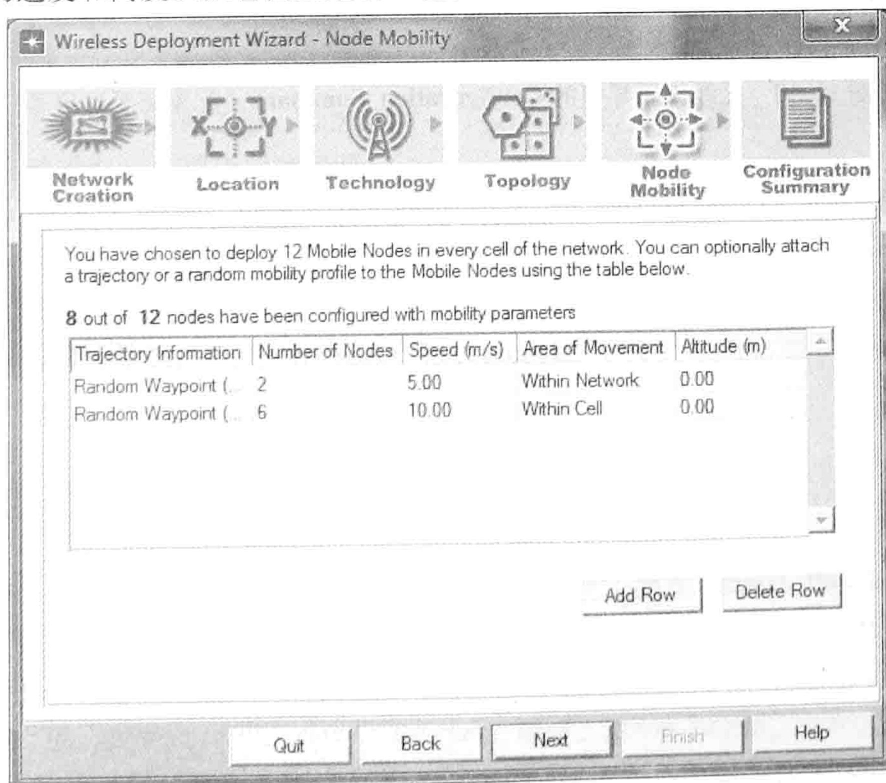


图 12.17　无线部署导引的 **Node Mobility**（节点移动性）配置屏幕

9）最后一个配置屏幕（见图 12.18）提供了所创建网络拓扑的一个简短摘要，并使您将所创建的网络保存到一个文件，以备将来重用。无线网络拓扑文件作为一个 XML 文件保存到默认模型目录中。但是，如果需要的话，可将文件保存到另外的位置。

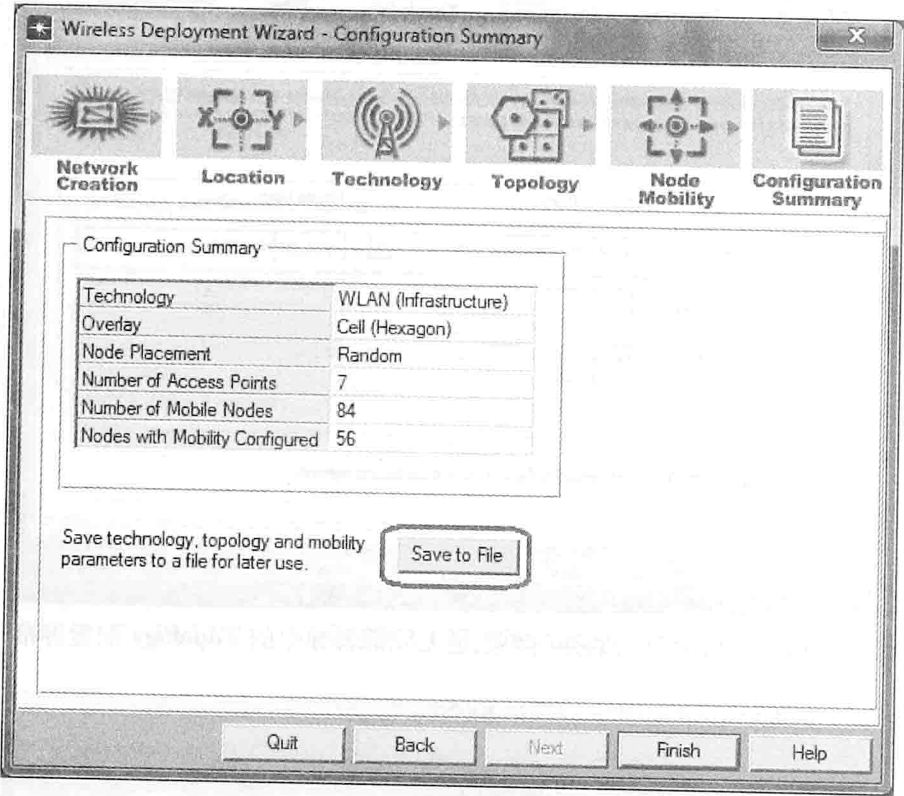


图 12.18　无线部署导引的 **Configuration Summary**（配置摘要）屏幕

实验室作业1：OPNET 引言

为完成这项实验室作业，建议阅读第1~4章。

L1.1 引言

在这项实验室作业中，将实施第一个完备的仿真研究。本实验室作业的目标是提供一项练习，该练习将使您可对 OPNET 软件的各种基本功能进行实践操作。具体而言，在第一个实验室作业中，将实施如下基本 OPNET 操作，如建立一个 OPNET 仿真项目、创建网络拓扑、配置个体对象、收集仿真统计量、运行仿真、管理场景和比较所收集的统计量。如果熟悉这些功能，那么可跳过这项作业。

在这项介绍性的实验室作业中，包括附加的配置细节，帮助新手 OPNET 用户通过创建其第一个 OPNET 仿真的各项挑战。在后面的各项实验室作业中，假定已经熟悉 OPNET 软件的基本功能，因此将略去许多这样的细节。相反，在后续作业中，将引导您去参考本书中主要各章的有关章节，在那里描述了相应的配置步骤。

L1.2 创建仿真项目和场景

在实施这个配置步骤之前，考虑复习一下1.6.2节，那里描述了如何使用 **Startup Wizard**（启动导引）创建项目和场景。

使用 **Startup Wizard**，创建名字分别为 Assignment 01 和 Initial_ Network 的一个新项目和一个空场景，即在 Initial Topology（初始拓扑）窗口内的选项 Create Empty Scenarios（创建空场景）。当配置初始场景设置时，在 **Network Scale**（网络规模）窗口中选择选项 Logical（逻辑的），并在 **Select Technologies**（选择技术）窗口中选择 *internet_ toolbox* 模型族。

L1.3 创建网络拓扑

建议您复习一下2.3节和2.4节，在那里分别描述了操作 **Object Palette Tree** 和创建网络拓扑的步骤。对于完成这项实验室作业接下来的部分，这些步骤是必要的。

第一步是创建如图 L1.1 所示的网络拓扑，方法是将如下对象模型拖入项目工作空间：*ethernet_ ip_ station*、*ip32_ cloud*、*ethernet_ slip8_ gtway* 和 *ppp_ server*。注意，**Object Palette Tree** 的默认视图不包含对象 *ethernet_ ip_ station*。需要搜索 **Object Palette Tree**，定位这个对象所属的模型族（见2.3.2节和2.4.1节）。

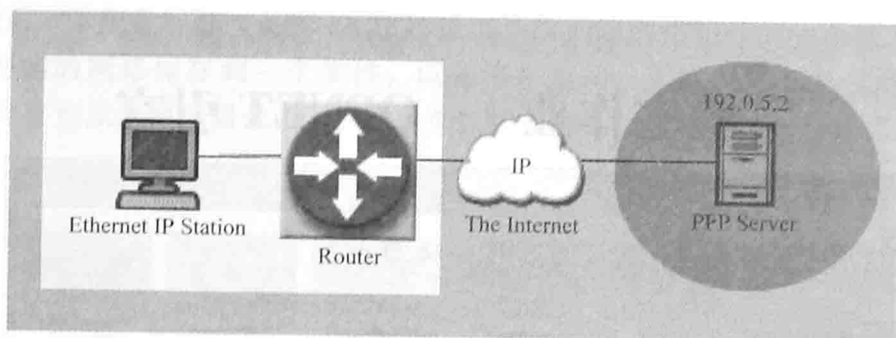


图 L1.1　这项作业的网络拓扑

一旦期望的节点模型被放到项目内，则需要使用链路模型连接它们。具体而言，使用 10*BaseT_ int* 链路模型连接 *ethernet_ ip_ station* 和 *ethernet_ slip8_ gtway*，使用 *PPP_ 56K* 链路模型连接 *ethernet_ slip8_ gtway* 和 *ip32_ cloud*，使用 PPP_ DS1 链路模型连接 ip32_ cloud 和 ppp_ server（见 2.4.2 节）。

在创建基本网络拓扑之后，验证您正确地连接了仿真网络中的各节点。参见 2.6.2 节了解验证链路连通性配置步骤的详细描述。

L1.4　配置网络拓扑

一旦创建了基本网络拓扑并验证了链路连通性，就可开始配置个体节点。具体而言，将需要实施如下步骤：

1）命名被仿真网络中的所有节点对象。通过分别在 3.2.2 节和 3.3 节描述的选择 **Set Name** 选项或改变节点属性 **name** 的值，可改变一个对象的名字。命名如图 L1.1 所示网络中的所有节点对象。

2）通过在 The Internet 节点中分别设置属性 **Performance Metrics... Packet Discard Ratio**（性能度量... 报文丢弃比率）和 **Performance Metrics... Packet Latency (secs)**［性能度量... 报文延迟（秒）］的值为 1.0% 和 50ms，配置通过互联网传输的报文所经历的丢失和延迟。

3）通过分别设置属性 **Address** 和 **Subnet Mask** 为 192.0.5.2 和 255.255.255.0，在节点 PPP_ server 中指定 IP 地址和子网掩码值。属性 **Address** 和 **Subnet Mask** 是复合属性 **IP... IP Host Parameters... Interface Information**（IP... IP 主机参数... 接口信息）的组成部分。

4）通过设置属性 **IP... Traffic Generation Parameters**（IP... 流量产生参数）为如下值，配置 Ethernet IP Station 节点中的流量产生源：

① **Packet Inter-Arrival Time (seconds)** = constant (0.5)

② **Packet Size (bytes)** = constant (2000)

③ **Destination IP Address** = 192.0.5.2

④ **Start Time (seconds)** = 100.0

注意，属性 **IP... Traffic Generation Parameters** 可由多个实例组成（见 3.3.3 节），即每行代表一个独立的流量源。在这个仿真场景中，仅关注于源自 Ethernet IP Station 节点的单个流量源，因此将仅需要一个属性实例。

5）创建被仿真网络的注释。打开 **Annotation Palette**（见 2.9 节），并在微软 Ethernet IP Station 和 Router 节点周围放置一个填充的矩形。同样在 PPP Server 节点周围放置一个填充的圆形，并在圆形内放置一个文本对象，该对象包含字符串 192.0.5.2，如图 L1.1 所示。可为每个添加的对象设置一个填充颜色，例如矩形为黄色，圆形为蓝色。

这就完成了创建网络仿真模型的过程。表 L1.1 中给出了网络配置的摘要。

表 L1.1 网络配置摘要

对象名字	对象模型	配置细节
Ethernet IP Station 节点	*ethernet_ ip_ station* 节点对象	属性 **IP... Traffic Generation Parameters** • **Packet Inter - Arrival Time (seconds) = constant (0.5)** • **Packet Size (bytes)** = constant (2000) • **Destination IP Address** = 192.0.5.2 • **Start Time (seconds)** = 100.0
Router 节点	*ethernet_ slip8_ gtwy* 节点对象	NONE
The Internet 节点	*ip32_ cloud* 节点对象	属性 **Performance Metrics** **Packet Discard Ratio** = 1.0% **Packet Latency (secs)** = 0.05
PPP Server 节点	*ppp_ server* 节点对象	属性 **IP... IP Host Parameters... Interface Information** • **Address** = 192.0.5.2 • **Subnet Mask** = 255.255.255.0
Ethernet IP Station < - > Router 链路	*10BaseT_ int* 链路对象	NONE
Router < - > The Internet 链路	*PPP_ 56K* 链路对象	NONE
The Internet < - > PPP Server 链路	*PPP_ DS1* 链路对象	NONE

L1.5 配置和运行仿真

在实验室作业的本节，将仿真配置为收集期望的统计量，并运行开发的仿真模型。这里第一步配置仿真收集如下统计量：

1. 类 IP 中的全局统计量（见 4.2.4 节）

Traffic Dropped (packets/sec)

2. 类 IP 中的节点统计项，要针对所有节点单独收集（见 4.2.3 节）

- **Traffic Dropped（packets/sec）**
- **Traffic Received（packets/sec）**
- **Traffic Sent（packets/sec）**

3. 类 point – to – point 中的链路统计量（见 4.2.3 节）

- **Throughput（bits/sec） – >**
- **Throughput（bits/sec） < –**
- **Throughput（packets/sec） – >**
- **Throughput（packets/sec） < –**

接下来，需要这样配置仿真（见 4.3 节）：

- **Duration** = 500 seconds
- **Seed** = 128
- **Values per statistic** = 100

现在执行仿真（见 4.3.7 节）。当执行完成时，检查产生的 DES 日志表项数（见 4.8 节）。如果日志表项总数小于 5，那么可能的情况是，仿真被正确配置，可继续到下一步，检查收集的结果。否则，打开 DES 日志，并检查由仿真产生的日志表项。在识别并改正任何配置错误之后，重新运行仿真。

L1.6　详细研究收集的结果

最后，详细研究在仿真过程中收集的统计量（见 4.5.1 节）。在作业的这部分，将显示 4 个仿真统计结果集合。第一个结果集合给出在网络中每个节点处丢弃的 IP 流量总量。第二个集合详细研究 Ethernet IP Station 发送的流量、由 PPP Server 接收的流量和由 The Internet 丢弃的流量之间的关系。第三个集合比较 Ethernet IP Station 节点发送的流量及在 Ethernet IP Station 和 Router 节点之间链路上转发的以报文数每秒为单位表示的流量速率。最后，也将详细研究 Ethernet IP Station 和 Router 节点之间链路上以比特每秒为单位表示的吞吐量。

表 L1.2 给出需要显示的统计结果的一个摘要。在独立分析平板中显示每个统计量集合。使用收集的结果回答如下问题：

表 L1.2　统计图形的配置摘要

集合	统计量	图形平板配置
1	针对如下统计类型的 **IP... Traffic Dropped（packets/sec）**： • 全局统计量 • Ethernet IP Station 节点统计量 • Router 节点统计量 • The Internet 节点统计量 • PPP Server 节点统计量	选项： • Stacked Statistics（堆叠的统计量） • As Is（保持原状）

（续）

集合	统计量	图形平板配置
2	Ethernet IP Stations 节点统计量 **IP... Traffic Sent （packets/sec）** PPP Server 节点统计量 **IP... Traffic Received （packets/sec）** The Internet 节点统计量 **IP... Traffic Dropped （packets/sec）**	选项： • Overlaid Statistics（覆盖式统计量） • As Is（保持原状）
3	Ethernet IP Station 节点统计量 **IP... Traffic Sent （packets/sec）** 链路 Ethernet IP Stations < - > Router 统计量 **point - to - point... throughput （packets/sec） - >**	选项： • Overlaid Statistics（覆盖式统计量） • As Is（保持原状）
4	链路 Ethernet IP Station < - > Router 统计量 **point - to - point... throughput （bits/sec） - >**	选项： As Is（保持原状）

问题

Q1. 详细研究显示统计量集合1的分析平板。

1）被仿真网络中的哪些节点丢弃报文？在被仿真网络中的哪些节点不丢弃任何报文？为什么？（提示：温习一下仿真的网络配置）。

2）全局统计量和节点统计量之间的区别是什么？对于丢弃报文的全局统计量和节点统计量，您观察到 **IP... Traffic Dropped （packets/sec）** 图形之间的任何区别了吗？为什么有区别或为什么没有区别？

Q2. 详细研究显示统计集合2的图形平板。

相比于由 PPP Server 接收到的流量和由 The Internet 丢弃的流量，Ethernet IP Station 发送的流量总量是什么样的？请解释。

Q3. 详细研究显示统计集合3的图形平板。

1）以报文数每秒为单位由仿真报告的 Ethernet IP Station 发送速率是多少？以报文数每 s 为单位，由仿真报告的 Ethernet IP Station 和 Router 之间链路上的吞吐速率是多少？这些值相同还是不同？为什么？

2）依据流量产生源配置，Ethernet IP Station 每0.5s产生一条报文，这对应于每 s2 个报文的发送速率。以报文数每 s 为单位，由仿真报告的 Ethernet IP Station 发送速率是多少？这些数值相同还是不同？为什么？就在 Ethernet IP Station 处发生的情况，您的假定是什么？（提示：由 Ethernet IP Station 产生的报文尺寸是多大和以太网中的 MTU 尺寸是多大？当高层报文太大以致不能在物理媒介上发送时，会发生什么情况？）

Q4. 详细研究显示统计集合4的图形平板。

以比特每秒为单位，在 Ethernet IP Station 和 Router 之间的链路上的发送速率是多少？由仿真报告的结果对应于流量源配置吗？（提示：假定 IP 首部和以太网首部分别为

20 字节和 26 字节）。

L1.7 复制场景，重新运行仿真并比较收集的结果

在作业的这部分，将创建当前场景的三个备份（见 1.7 节），并如下修改场景配置：

1）复制 Initial_ Network 场景，命名新场景为 Packet_ Size_ 1000bytes。在 Ethernet IP Station 节点中修改流量产生源的配置，从而使每条新产生的报文为 1000 字节而不是 2000 字节。先不要执行仿真。

2）复制 Initial_ Network 场景，命名新场景为 500_ values_ statistic。通过设置属性 Values per statistic 为 500，修改仿真配置。在没有运行仿真之前保存仿真配置（见 4.3 节）。

3）复制 Initial_ Network 场景，命名新场景为 5000_ values_ statistic。通过设置属性 **Values per statistic** 为 5000，修改仿真配置。在没有运行仿真之前保存仿真配置（见 4.3 节）。

使用 **Manage Scenarios**（管理场景）菜单（见 1.7.2 节），为 Assignment 01 项目内的所有场景（重新）收集统计量。一旦执行了所有场景，如下显示所收集的统计量：

1）在单个图形平板中使用 *Overlaid Statistics* 和 *As Is* 选项，为场景 Initial_ Network 和 Packet_ Size_ 1000bytes 显示 Ethernet IP Station 节点统计量 **IP ... Traffic Sent (packets/sec)**（见 4.4 节）。

2）在单个图形平板中使用 *Overlaid Statistics* 和 *As Is* 选项，为场景 Initial_ Network 和 500 _ values _ statistic 和 5000 _ values _ statistic 显示 The Internet 节点统计量 **IP... Traffic Dropped (packets/sec)**（见 4.4 节）。

现在，回答如下问题：

问题

Q5. 在第一幅图中您观察到什么，该图显示由 Ethernet IP Station 节点发送的流量？在 Initial_ Network 和 Packet_ Size_ 1000bytes 场景中收集的结果所报告的报文发送速率方面，有什么差异吗？为什么？注意您没有在节点 Ethernet IP Station 处改变报文到达间隔速率。这些结果确认还是反驳了在 L1.6 节为问题 3 的 2）形成的假设。

Q6. 在第二幅图中您观察到什么，该图显示由 The Internet 节点丢弃的流量？在每个场景中丢弃的报文数量是相同还是不同？为什么相同或不同？［提示：回忆一下统计量是如何在 OPNET 中收集的和属性 **Values per statistic** 如何影响所报告的结果的（见 4.1.1 节和 4.2.7 节）］。

实验室作业 2：简单容量规划

为完成这项实验室作业，建议阅读第 1 ~4 章。

L2.1 引言

在本实验室作业中，将详细研究链路容量对网络性能的影响，练习配置流量需求并为单个属性指定多个值。具体而言，将在各种条件下评估给定网络的性能，并推荐一个最优配置。

考虑如下情况：ABC 无限责任公司，这是一个小型的私营公司，正处在扩张过程，希望在城镇另一侧增加另一处办事处。该公司计划在未来将新办事处的规模翻倍。在这项实验室作业中，将帮助 ABC 公司确定准备链路（将其办事处连接到互联网）的最佳选项。

L2.2 对 ABC 公司的网络建模

创建名字分别为 Assignment 02 和 ABC_ Network 的一个新项目和空场景（见 1.6.2 节）。创建如图 L2.1 所示的网络拓扑。确保使用在表 L2.1 中指定的节点和链路模型（遵循 2.4 节的指令）。一旦创建了网络拓扑，验证链路连通性（见 2.6.2 节）。

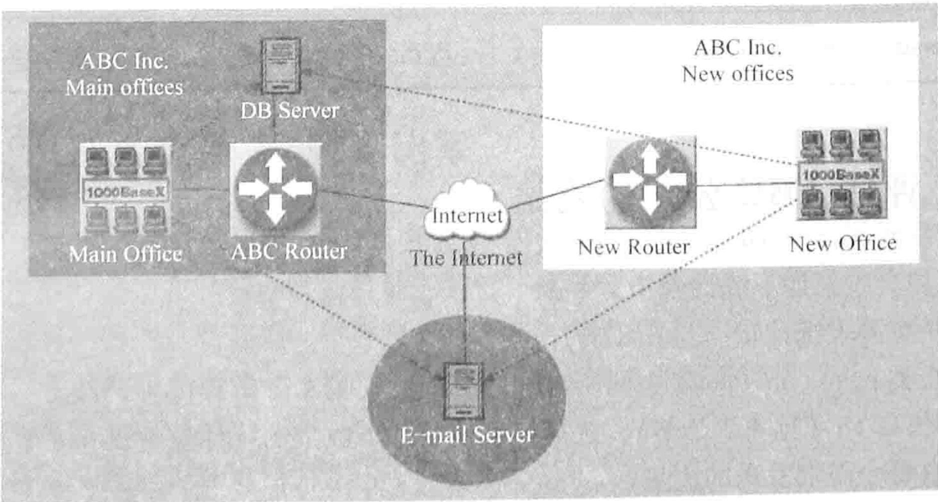


图 L2.1　这项作业的网络拓扑

接下来，需要添加和配置流量需求（见 6.6 节）。如下添加和配置四个 *ip_ traffic_ flow* 需求模型：

1）所有需求应该配置为有1%的流量建模为显式流量（属性 **Traffic Mix**）。

2）所有需求应该在时刻100s处开始传输数据。

3）直到仿真结束之前，所有需求应该继续传输数据。

4）Main Office→DB Server 和 New Office→DB Server 这两个 IP 需求都以1200kbit/s 和100报文数/s的恒定速率传输数据。

5）Main Office→E - mail Server 和 New Office→E - mai Server 这两个 IP 需求都以800kbit/s 和10报文数/s的恒定速率传输数据。

表 L2.1 ABC 公司的网络拓扑摘要

对象名	对象模型
DB Server	*ethernet_ server* 节点对象
E - mail Server	*ppp_ server* 节点对象
Main Office New Office	1000*BaseX_ LAN* 节点对象
ABC Router New Router	*ethernet4_ slip8_ gtwy* 节点对象
The Internet	*ip32_ cloud* 节点对象
Main Office < - > ABC Router DB Server < - > ABC Router New Office < - > New Router	1000*BaseX* 链路对象
ABC Router < - > The Internet New Router < - > The Internet	*PPP_ DS1_ int* 链路对象
The Internet < - > E - mail Server	*PPP_ DS3* 链路对象

L2.3 评估 ABC 公司的网络

配置仿真收集如下统计量：

1）所有需求统计量（见6.8节）。

2）在类 point - to - point 中的所有链路统计量（见4.2.3节）。

执行仿真1h（见4.3.7节），之后详细研究如下收集的统计量（见4.5节）：

1）由每个需求发送的流量。

2）在连接到 The Internet 节点的所有链路上的利用率。

3）由每个需求的目的地接收到的流量。

4）由需求报文经历的端到端延迟。注意，如果没有配置需求显式产生流量（即没有将 **Traffic Mix** 属性设置为大于0%的一个值），那么这个需求统计量将不报告数据。

问题

Q1. 每个需求的发送速率是多少？仿真结果对应于需求配置吗？

Q2. 在连接到互联网的链路上的利用率是多少？为什么？（提示：1000Base - T、DS - 1 和 DS - 3 链路的容量是多少？每个需求的传输速率是多少？）

Q3. 流量到达每个需求的目的地的速率是多少？为什么？

Q4. 对于每个需求，报文端到端延迟是多少？为什么？

Q5. 以您的意见，当前网络配置的问题是什么？网络中的哪条链路是瓶颈？

L2.4 比较应用性能

复制原场景，并创建一个称为 Eliminating Bottleneck 的新场景（见 1.7 节）。在新场景中，将瓶颈链路的容量设置为 **smallest**（最小）值，这将导致瓶颈链路利用率在 60% 左右。注意，如果链路对象不包含属性 **data rate**（代表链路的容量），那么需要将链路模型改变为在其名字中包含扩展名_ *int* 的一个链路模型。为改变与对象相关联的模型，首先需要在对象的 **Edit Attributes** 窗口中选择 *Advanced* 检查框，之后改变属性 **model** 的值（见 3.4.5 节）。在更新瓶颈链路的容量之后，重新运行仿真，并详细研究所收集的统计量。

问题

Q6. 瓶颈链路的新容量应该是多少？

Q7. 连接到互联网的链路上的利用率是多少？为什么？

Q8. 流量到达每个需求的目的地的速率是多少？为什么？

Q9. 每个需求的流量所经历的端到端延迟是多少？为什么？

L2.5 识别最优带宽/成本比率

复制场景 Eliminating Bottleneck，并创建称为 Best Option 的一个新场景（见 1.7 节）。假定 ABC 公司在其新办事处将员工翻倍，这导致由这些办事处产生的流量总量翻倍。为对这样的行为进行建模，将源自 New Office 节点的流量需求的传输速率翻倍。

当前，互联网服务提供商（ISP）为连接到互联网提供如下选项：

1）T1 线路，$50 每月。

2）4Mbit/s 线路，$100 每月。

3）10Mbit/s 线路，$150 每月。

4）50Mbit/s 线路，$500 每月。

将瓶颈链路上的属性 **data rate** 提升，之后将被提升属性的值设置为通过 ISP 可用的互联网接入速率［即 T1 或 DS1、4Mbit/s、10Mbit/s 和 50Mbit/s（见 3.5 节）］。重新

运行仿真，并详细研究瓶颈链路利用率和 IP 需求的报文（源自 New Office 节点）所经历的端到端时延。

问题

当回答如下问题时，考虑在瓶颈链路上指定的四种不同容量。

Q10. 在瓶颈链路上的链路利用率是多少？在 New Office 和 E - mail Server 之间 IP 需求流量的报文端到端时延是多少？基于这些结果，ABC 公司应该选择哪种链路容量将其新办事处连接到互联网？

Q11. 在 New Office 和 DB Server 之间 IP 需求流量的报文端到端时延是多少？为什么？

实验室作业 3：标准应用介绍

针对这项实验室作业，建议阅读第 1 ~5 章和第 7 章。

L3.1 引言

这项实验室作业的主要目标是介绍 OPNET 中标准应用的建模。具体而言，在这项作业中，将配置几个标准应用，指定和调试所创建的用户概要定义，在网络中部署指定的应用，最后是详细研究应用性能。

考虑如下情形，其中称为研究人员无限责任公司（Researchers Inc.）的一家 R&D 公司分散在三个地点：

1）Main Site（主地点）由 50 名雇员组成，主要使用电子邮件、网页浏览和话音会议实施他们的日常工作。

2）RnD Site（研发地点）由 100 名雇员组成，主要使用远程登录、电子邮件和网页搜索实施他们的日常任务。

3）Server Farm（服务器集群）包含用于电子邮件、网站和远程登录应用的服务器。

作业是创建研究人员公司网络的一个仿真模型，并详细研究部署在网络中的各项应用的性能。

L3.2 对研究人员公司的网络拓扑进行建模

在实验室作业的这部分，将通过实施如下步骤创建研究人员公司网络的一个模型：

1）创建名字分别为 Assignment 03 和 Researchers Network 的一个新项目和一个空场景（见 1.6.2 节）。

2）创建如图 L3.1 所示的网络拓扑。确保使用的是表 L3.1 中指定的节点和链路模型（遵循 2.4 节中的指令）。

3）配置 Main Site 和 RnD Site 的 LAN 对象分别包含 50 个和 100 个工作站。通过设置属性 **LAN... Number of Workstations**（LAN... 工作站数量）为期望的值，可指定一个 LAN 对象中工作站的数量。

4）一旦创建了网络拓扑并配置了 LAN 对象，验证链路连通性（见 2.6.2 节）。

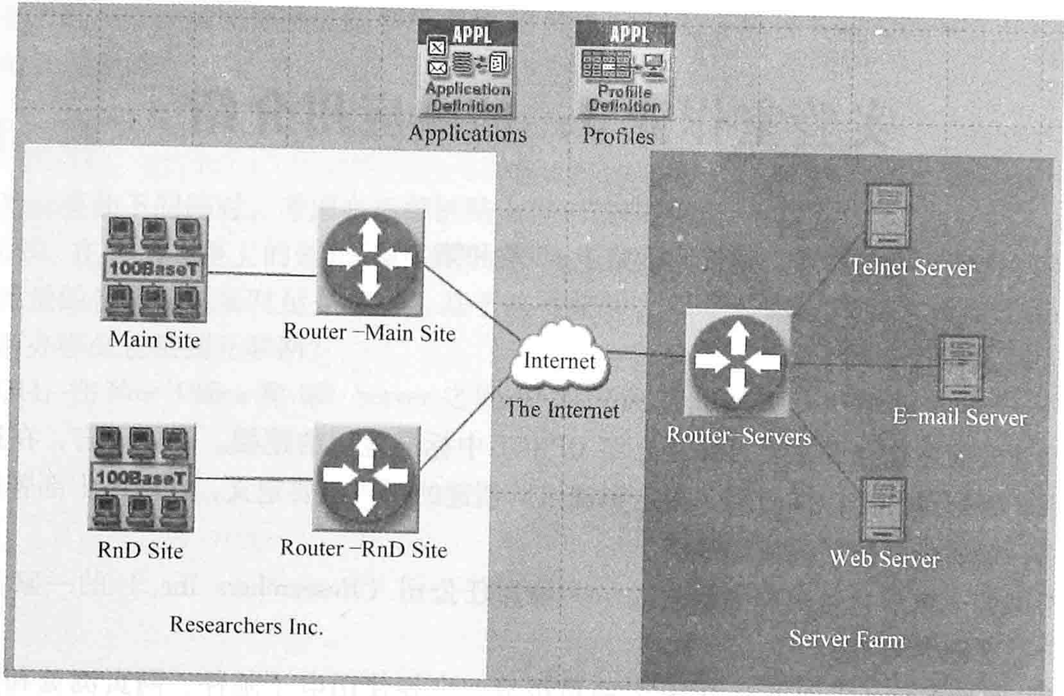


图 L3. 1 这项作业的网络拓扑

表 L3. 1 研究人员公司网络拓扑的摘要

对象名	对象模型
Telnet Server E – mail Server Web Server	*ethernet_ server* 节点对象
Router – Servers Router – Main Site Router – RnD Site	*ethernet4_ slip8_ gtwy* 节点对象
Main Site RnD Site	100*BaseT_ LAN* 节点对象
The Internet	*ip32_ cloud* 节点对象
Main Site < – > Router – Main Site RnD Site < – > Router – RnD Site Telnet Server < – > Router – Servers E – mail Server < – > Router – Servers Web Server < – > Router – Servers	100*BaseT* 链路对象
The Internet < – > Router – Main Site The Internet < – > Router – RnD Site The Internet < – > Router – Servers	*PPP_ DS1_ int* 链路对象

L3.3 在研究人员公司网络中配置和部署应用

在实验室作业的这部分，通过实施如下步骤，将在研究人员公司网络中配置和部署应用：

1）配置个体应用（见5.3.2节）。

① 将 **Application Config** 节点添加到项目工作空间中（如果还没有添加的话）。

② 依据表L3.2中提供的配置细节，定义5个标准应用。

2）定义用户概要（见7.2节）。现在，假定每名研究人员公司雇员都顺序地（即一个接着一个）执行应用。基于这个假定，将创建两个概要：

① Main Site Employees，运行如下三项应用：Web Browsing、E－mail和VoIP。

② RnD Employees，运行如下三项应用：Web Research、Telnet和E－mail。

3）配置每项概要都以Serial（Ordered）**Operation Mode**运行应用，而将其他概要配置属性保留设置为其默认值。

4）依据如下规则，部署所创建的概要（见7.4节）：

① Main Site Employee概要被配置为以Main Site作为概要的源，而对象Web Server、E－mail Server和RnD Site分别作为Web Browsing、E－mail和VoIP应用的服务器或目的地。

② RnD Employees概要被配置为以RnD Site作为概要的源，而对象Web Server、Telnet Server和Email Server分别作为Web Research、Telnet和E－mail应用的服务器或目的地。

5）配置两个LAN对象使所部属的概要在LAN内的所有节点上执行。通过设置属性 **Applications...Application：Supported Profiles...<profile>...Number of Clients** 为Entire LAN，可指定这项要求，其中 **<profile>** 是 **Main Site** 或 **RnD Employee**。

6）为 **Email**、**HTTP**、**Remote Login** 和 **Voice** 应用，配置仿真收集所有全局统计量（见4.2.4节和5.6节）。

7）执行仿真5min（见4.3节）。

表L3.2 应用配置设置

应用名字	应用模型属性	应用模型属性值
Web Browsing	**Http**	Heavy Browsing
Telnet	**Remote Login**	High Load
Web Research	**Http**	Searching
E－mail	**Email**	Medium Load
VoIP	**Voice**	GSM Quality Speech

问题

Q1. 在仿真完成时产生了多少DES日志表项（见4.8节）？

Q2. 打开**Log Viewer**窗口（**DES→Open DES Log**）并详细研究处理应用建立的DES日志表项（见4.8节）。由那个日志表项报告的问题是什么？

Q3. 针对每项应用，详细研究全局统计量**Traffic Sent (bytes/sec)**。仿真结果是确认还是反驳了DES日志告警？为什么确认或为什么反驳（即原因是什么）？

Q4. 我们原假定和后续步骤中的哪个导致这个问题产生？如何纠正这个问题？

L3.4 改正第一个配置问题

改正上述问题的一种方式是改变在概要内如何执行各应用。当前配置仅允许概要内的第一个应用启动。其他应用从来就不会执行，原因是第一个应用配置为直到仿真结束之前都运行，而概要内各应用是以Serial（Ordered）［串行（顺序的）］方式执行的，这意味着下一个应用仅在前一个应用完成其执行之后才启动（见7.6节）。

明显的是，我们的原假定，即各应用一个接一个地执行，是不正确的。实际上，常见情况是，一名用户同时运行网页浏览、电子邮件、Telnet和其他应用，需要时在应用之间切换。为对这种行为建模，实施如下步骤：

1）创建称为Fixing Problem 01的一个复制场景（见1.7节）。

2）对于Main Site Employee和RnD Employee概要，将工作模式从Serial（Ordered）改变为Simultaneous（见7.2节）。

3）配置仿真也要收集链路*point-to-point*统计量（见4.2.3节）。

4）重新运行仿真。

问题

Q5. 详细研究每个应用的发送速率。上述概要配置改变解决了在前一节发现的问题了吗？

Q6. 详细研究电子邮件（E-mail）应用的下载响应时间、网页（Web）应用的页面响应时间、Telnet应用的响应时间和VoIP报文所经历的端到端时延。使用平均（average）函数在单个分析平板中显示这些统计量（见4.5节）。所显示的应用响应时间和时延图形展现出一个趋势吗？为什么有趋势或为什么没有？

Q7. 您认为应用所经历的时延对公司雇员是可接受的还是不可接受的？为什么可接受或为什么不可接受？

Q8. 详细研究将网络路由器连接到互联网的链路上的利用率。依据所观察到的链路利用率，哪个应用导致网络中的拥塞？

L3.5 改正第二个配置问题

在前一节中收集的仿真统计量报告，链路Router-Main Site <->The Internet和

Router - RnD Site < - >The Internet 的利用率为100%，而 Internet 和 Server farm 之间的链路利用率小于20%。这提示 VoIP 应用是导致网络拥塞的罪魁祸首。在与研究人员公司员工的进一步讨论之后，发现 Main Site 实际上一天仅建立一次话音会议。通常情况下，这种会议呼叫仅持续大约1h，并在下午大约2：00PM 在单个会议室进行。

为对这种行为建模，将需要修改应用部署，改变 Main Site Employee 概要的定义，并创建一个附加的概要，对会议呼叫建模。具体而言，需要实施如下配置步骤：

1）复制当前场景 Fixing Problem 01，创建称为 Fixing Problem 02 的一个新场景（见1.7节）。

2）清除在网络中的应用部署（见7.4.8节）。

3）从 Main Site Employee 概要的定义中去除 VoIP 应用（见7.2节）。

4）创建称为 Conference Call 的一个新概要，它仅运行 VoIP 应用。VoIP 应用应该在概要启动之后6h启动。它应该仅运行1h，且它应该重复一次。可将其他属性设置保持为其默认值。注意，配置属性接受以秒为单位表示的时间值，且概要的工作模式对仿真没有影响，原因是该概要由单个应用组成（见7.2节）。

5）在网络中重新部署所有概要。配置 Conference Call 概要，对于 VoIP 应用，以 Main Site 作为源，RnD Site 作为目的地（见7.4节）。

6）修改 Main Site LAN 对象的配置，使概要 Conference Call 运行在单个客户端上（即属性 **Applications... Application：Supported Profiles... Conference Call... Number of Clients**）。

7）将仿真时长设置为8h（见4.3节）。

8）重新运行仿真。

问题

Q9. 详细研究电子邮件应用的下载响应时间、网页应用的页面响应时间、Telnet 应用的响应时间和 VoIP 报文所经历的端到端时延。相比于在场景 Fixing Problem 01 中收集的结果，这些值情况如何？您认为这种时延对应用用户来说是可接受的吗？为什么可接受或为什么不可接受？

Q10. 详细研究每个应用的发送速率。相比于在场景 Fixing Problem 01 中收集的结果，这些值情况如何？哪项应用产生最多的流量？通过应用配置，验证观察到的结果（即详细研究每个应用设置，并计算由每个应用产生的数据量）。

Q11. 详细研究将网络路由器连接到互联网的链路上的利用率。新的概要配置去除了网络中的瓶颈了吗？

实验室作业4：HTTP性能

针对这项实验室作业，建议阅读第1~5章和第7章。

L4.1 引言

在这项实验室作业中，将详细研究HTTP各种版本的性能。特别地，将研究HTTP的持久变种、非持久变种在带有流水线和没有流水线情况下对一项Web应用性能的影响。

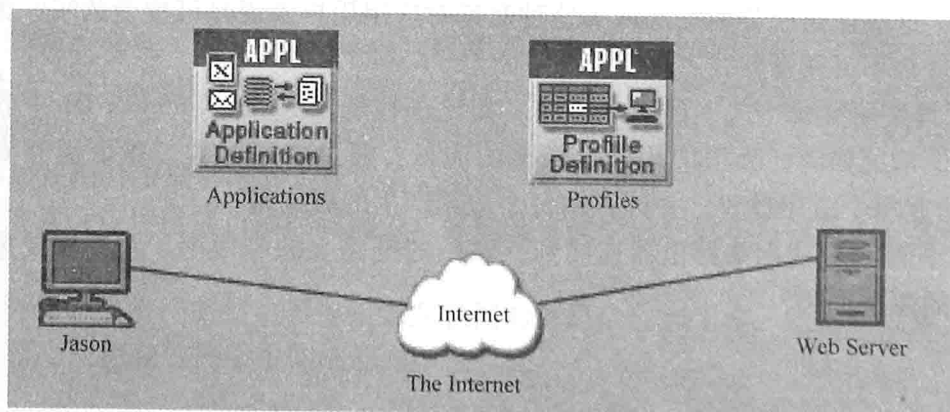
考虑如下场景：Jason是一名硕士生，他在互联网上实施研究。Jason的计算机通过一条T1线路连接到互联网。Jason使用运行HTTP版本1.1的一个网页浏览器。Jason的日常工作日看起来如下：Jason来到他的办公室，启动他的计算机，并登录到网络，这用去大约100s。之后Jason运行他的网页浏览器，启动大约用去5~10s。通常情况下，Jason每隔10s访问一个新的页面。一般而言，每个网页由几幅大型图像组成。Jason访问的多数网页驻留在同一台服务器上。在2h之后Jason停止浏览。

L4.2 创建仿真模型

在作业的这部分，通过实施如下步骤，将创建Jason网络的一个仿真模型：

1）分别创建名为Assignment 04和HTTP_1_1的一个新项目和一个空场景（见1.6.2节）。

2）使用在表L4.1中指定的节点和链路模型，创建如图L4.1所示的网络拓扑。



图L4.1 这项作业的网络拓扑

3）配置The Internet云丢弃所有到达报文的1.0%，并对所有通过它的流量引入一

个附加的 200ms 时延（参见实验室作业 1 了解配置细节）。

4）验证链路连通性（见 2.6.2 节）。

5）如下配置 Web 应用（见 5.3 节）：

① 命名 Jason 的网页浏览应用为 Web。

② 配置 Jason 的应用运行在 HTTP 1.1 之上［通过编辑属性 **HTTP Specification**（见 5.4.4 节）］。

③ 配置 Jason 的应用大约每隔 10s 请求一个新的网页。使用指数分布来仿真这样的行为，通过属性 **Page Interarrival Time（seconds）**。

表 L4.1　网络拓扑摘要

对象名	对象模型
Jason	*ppp_ wkstn* 节点对象
Web Server	*ppp_ server* 节点对象
The Internet	*ip32_ cloud* 节点对象
Jason < - > The Internet Web Server < - > The Internet	*PPP_ DS1_ int* 链路对象

④ 配置应用访问网页，每个网页由尺寸为 100 字节的单个文本对象和 7 个内嵌于文本对象内的大型图像组成（使用复合属性 **Page Properties**）。

⑤ 保持其他属性设置为其默认值。

6）创建称为 Web Research 的一个概要，表示如 L4.1 节所述的 Jason 的行为（见 7.2 节）：

① Jason 仅在其计算机上运行网页浏览器（即概要仅包含称为 Web 的一项应用）。

② 在 Jason 登录到计算机之后 5 ~ 10s，他启动网页浏览器（即在概要启动之后 5 ~ 10s 应用启动）。使用均匀分布对应用启动时间偏移进行建模。

③ Jason 浏览网页 2h（即应用运行 2h）。使用常数分布对应用的时长进行建模。

④ 启动计算机用去 100 ~ 110s（即在仿真启动之后 100 ~ 110s 概要启动）。使用均匀分布对概要启动时间建模。

⑤ 一旦 Jason 关闭计算机则仿真终止（即直到仿真结束之前，概要一直运行）。

7）部署概要 Web Research，从而节点 Jason 是概要的源，节点 Web Server 是 Web 应用的服务器（见 7.4 节）。

8）配置仿真收集如下统计量：

① HTTP 应用的所有全局统计量（见 4.2.4 节和 5.6 节）。

② 在所有节点上来自类 **TCP** 的节点统计量 **Active Connection Count**（见 4.2.3 节）。

③ 可能感兴趣的任何其他统计量。

9）执行仿真 8000s（见 4.3 节）。

问题

Q1. 被报告的页面响应时间和对象响应时间是多少？如何比较这些值？为什么？

Q2. 详细研究在 HTTP 客户端和服务器节点上报告的活跃 TCP 连接最大数量。HTTP 版本 1.1 的哪个性质解释所观察到的行为？为什么？

L4.3 HTTP 1.0 和 HTTP 1.1

在实验室作业的这部分，将比较运行 HTTP 版本 1.0 和 HTTP 版本 1.1 时的应用性能。在创建一个新的仿真场景之前，就期望观察到的情况，写下假定。具体而言，在运行 HTTP 版本 1.0 而不是 HTTP 1.1 时，解释页面和对象响应时间和活跃 TCP 连接数将如何变化。验证假定，方法是创建另一个仿真场景，其中在 HTTP 1.0 之上运行网页浏览器，执行仿真场景，之后详细研究收集到的结果：

1）复制当前场景，命名新的场景为 HTTP_ 1_ 0（见 1.7 节）。

2）将 Web 应用改变为运行 HTTP 版本 1.0 而不是版本 1.1（见 5.4.4 节）。

3）执行仿真场景 HTTP_ 1_ 0（见 4.3 节）。

问题

如果正确地配置了仿真，那么应该观察到如下情况：

1）HTTP 版本 1.1 的页面响应时间将比 HTTP 版本 1.0 的页面响应时间小。

2）HTTP 版本 1.1 的对象响应时间将大于 HTTP 版本 1.0 的对象响应时间。

回顾一下 HTTP 1.1 和 HTTP 1.0 的规范，并解释观察到的行为（即可详细研究在 **HTTP** 应用的定义内为复合属性 **HTTP Specification** 指派的值）。具体而言，回答如下问题：

Q3. 详细研究在场景 HTTP_ 1_ 1 和 HTTP_ 1_ 0 内收集的页面响应时间和对象响应时间统计量。您观察到什么？为什么 HTTP 1.1 的页面响应时间小于 HTTP 1.0 的页面响应时间，而 HTTP 1.1 的对象响应时间大于 HTTP 1.0 的对象响应时间。

Q4. 详细研究在每个场景中由 Web 应用建立的活跃 TCP 连接数。HTTP 的哪个版本具有较少的活跃 TCP 连接？为什么？

注释：当详细研究由 HTTP 流量创建的活跃连接数时，使用在 Web Server 节点处收集的统计量，不要太关注于活跃 TCP 连接数的绝对值。被报告的统计值可能看起来是不正确的。这主要是由于 OPNET 收集统计量的机制本质以及一条 TCP 连接仍然保持打开状态的时间段长度的原因。每次一个节点打开或关闭一条 TCP 连接，OPNET 将相应统计量的值加 1 或减 1。在仿真层次，活跃 TCP 连接数统计量被记录为一段时间（典型地大于 1s）上统计量值的和。在客户端侧，在一个非常短的时间上该 TCP 连接保持打开状态。结果是，在客户端侧的打开连接数经常被记录为 0。另外，服务器侧在一段较

长的时间上保持 TCP 连接出于打开状态，这通常可由 OPNET 的统计量收集机制正确记录。另外，在服务器侧处理不同 HTTP 请求的 TCP 连接，在相同时间仍然是打开的，导致被报告的活跃 TCP 连接数大于被请求页面对象的活跃数。

L4.4 带有流水线和不带有流水线的 HTTP

在实验室作业的这部分，将详细研究 HTTP 流水线功能如何影响 Web 应用的性能。在做这部分作业之前，写下期望观察到什么。具体而言，解释当使用不带流水线的 HTTP 版本 1.1 时，页面响应时间和对象响应时间以及活跃 TCP 连接数量将如何改变。验证假定，方法是创建另一个仿真场景，其中在不带流水线的 HTTP 1.1 之上运行网页浏览器，执行该场景，之后详细研究所收集的结果：

1）复制场景 HTTP_ 1_ 1，将新的场景命名为 HTTP_ 1_ 1_ no_ pipelining（见 1.7 节）。

2）改变 Web 应用运行 HTTP 版本 1.1，将流水线请求总数设置为 1（见 5.4.4 节）。

3）执行仿真场景 HTTP_ 1_ 1_ no_ pipelining（见 4.3 节）。

4）对所有三个场景比较所收集的统计量。考虑使用 average（平均）选项，显示所收集的响应时间统计量（见 4.5.4 节）。

问题

如果正确地配置了仿真，那么应该观察到如下情况：

1）不带流水线的 HTTP 1.1 的对象响应时间将是最小的。

2）不带流水线的 HTTP 1.1 的页面响应时间将是最大的。

回顾一下持久 HTTP 连接和流水线的定义，并解释观察到的行为。具体而言，回答如下问题：

Q5. 为什么不带流水线的 HTTP 1.1 具有最小对象响应时间？

Q6. 为什么不带流水线的 HTTP 1.1 具有最大的页面响应时间？

Q7. 带流水线与不带流水线在 HTTP 版本 1.1 的情况下，活跃 TCP 连接数有什么区别？为什么？注意，为了回答这个问题，可能希望详细研究仅在客户端上收集的统计量。

L4.5 简单的网页

在这项实验室作业的最后部分，将详细研究不同的 HTTP 配置如何影响一个 Web 应用的性能，该应用仅请求简单的单对象网页。在开始这部分作业之前，就对象响应时间、页面响应时间，以及对于 HTTP 版本 1.0、带有流水线的 HTTP 版本 1.1 和不带流水线的 HTTP 版本 1.1、活跃 TCP 连接数这几方面写下假设。通过实施如下步骤验证假设：

1）复制场景 HTTP_ 1_ 1，将新场景命名为 HTTP_ 1_ 1_ simple（见 1.7 节）。

2）通过将网页设置为由单个 1000 字节对象组成的做法，改变 Web 应用定义（见 5.4.4 节）。

3）针对场景 HTTP_ 1_ 0 和 HTTP_ 1_ 1_ no_ pipelining，重复步骤 1）和步骤 2），分别命名新的场景为 HTTP_ 1_ 0_ simple 和 HTTP_ 1_ 1_ no_ pipelining_ simple。

4）执行新创建的场景（见 4.3 节）。

5）为所有这三个场景比较收集到的统计量。考虑使用 average 选项，显示所收集的统计量（见 4.5 节）。

问题

如果正确地配置了仿真，那么应该观察到如下情况：

1）HTTP 1.1（带有流水线和不带有流水线）的页面响应时间小于 HTTP 1.0 的页面响应时间。

2）在所有情形中，对象响应时间几乎是一样的。

回顾一下持久和非持久 HTTP 连接的配置设置和定义。通过回答如下问题，解释观察到的行为：

Q8. 当请求一个简单的网页时，带有流水线和不带有流水线的 HTTP 版本 1.1，在活跃 TCP 连接数上有何不同？为什么？

Q9. 为什么 HTTP 1.1 的页面响应时间小于 HTTP 1.0 的页面响应时间？

Q10. 为什么 HTTP 1.1 和 HTTP 1.0 的对象响应时间几乎相同？

实验室作业5：对定制应用建模

针对这项实验室作业，建议阅读第1～7章。

L5.1 引言

在这项实验室作业中，将开发两项定制应用，之后要比较它们的性能：

1）提供具有认证的加密通信的一项应用。

2）仅提供明文通信的一项应用。

特别地，将仔细研究增加的安全等级如何影响整体应用性能。

考虑如下场景：Alice为与Bob通信开发了一项应用。Alice希望在网络上安全地通信，但她不确信的是，增加的安全层是否因为引入太多的延迟而妨碍她的应用的性能。在开始程序开发之前，Alice希望实施一项仿真研究，比较其应用的两个可能版本的性能。其应用的第一个版本，称为Secure AB Talk（安全的AB交谈），由两个不同部分组成：认证和会话密钥发现，以及加密的消息交换。其应用的第二个版本，称为AB Talk（AB交谈），仅进行明文消息交换。

Secure AB Talk应用的认证和会话密钥发现部分工作过程如下描述。将采用密钥K加密的消息M表示为$K\{M\}$。

1）Alice向密钥分发中心（KDC）（一个被信任的第三方权威机构）发送一条消息，为其与Bob的通信请求一个会话密钥。

2）KDC为Alice和Bob之间的通信产生一个会话密钥K_{AB}。

3）KDC向Alice发送一条消息$K_{AP}\{K_{AB}, T_B\}$，其中：

① K_{AP}为Alice的公开密钥。

② T_B为由KDC产生的要发送到Bob的票据。$T_B = K_{BP}\{K_{AB}, \text{“}Alice\text{”}\}$。

③ K_{BP}为Bob的公开密钥。

④“*Alice*”为Alice的身份。

4）Alice从KDC接收上述消息，并使用她的私有密钥解密该消息，前提条件是Alice有K_{AB}和T_B。

5）Alice向Bob发送一条消息$\{T_B, K_{AB}\{S_A\}\}$，其中S_A是某个秘密数。

6）Bob从Alice接收消息，并使用他的私有密钥解密T_B，由此得到K_{AB}。

7）Bob使用检索得到的密钥K_{AB}解密Alice的秘密数S_A。这证明了Alice的身份，因为仅有知道Alice的私有密钥的某个人才可得到K_{AB}和T_B票据（在步骤4中包含Alice的身份）。

8）Bob 加密并向 Alice 发送一条消息 K_{AB} $\{S_A+1\}$。Bob 加密值 S_A+1，目的是防止重放攻击。

9）Alice 使用会话密钥 K_{AB} 解密 Bob 的消息，验证 Bob 的身份，因为仅有 Bob 知道值 S_A 和会话密钥 K_{AB}。

上面描述的算法是松散地基于 Kerberos 认证服务的，且图 L5.1 给出其可视图形（visual）概要。

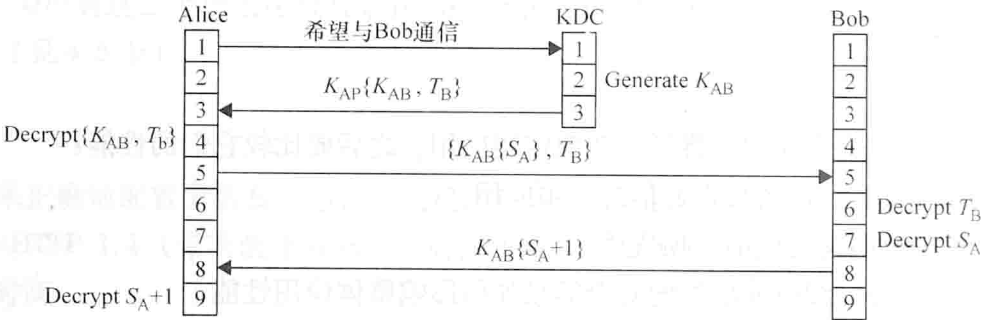


图 L5.1　Secure AB talk 认证协议

Secure AB Talk 应用的加密数据交换部分由 Alice 和 Bob 之间加密消息的异步交换组成。AB Talk 应用由单一部分组成，这对 Alice 和 Bob 之间明文消息的交换进行建模。明文数据交换也是异步完成的，这意味着在接收到一条请求之前，Alice 可向 Bob 发送一条消息，反之亦然。在这个模型中，假定加密并不改变正被传输消息的尺寸。但是，加密是一种计算密集的任务，因此将增加消息处理时间。

L5.2　创建仿真模型

在本作业的这部分，将创建由 Alice 和 Bob 通信所用网络的一个仿真模型。

1）创建一个新的项目和一个空的场景，分别命名为 Assignment 05 和 AB_ Talk（见 1.6.2 节）。

2）使用表 L5.1 中指定的节点和链路模型，创建如图 L5.2 所示的网络拓扑（见 2.3 节和 2.4 节）。确保为 Alice 和 Bob 节点使用高级模型（即扩展名_ *adv*）。

表 L5.1　网络拓扑摘要

对象名	对象模型
Alice，Bob	*ppp_ wkstn_ adv* 节点对象
KDC	*ppp_ server* 节点对象
The Internet	*ip32_ cloud* 节点对象
Alice < - > The Internet KDC < - > The Internet Bob < - > The Internet	*PPP_ DS1_ int* 链路对象

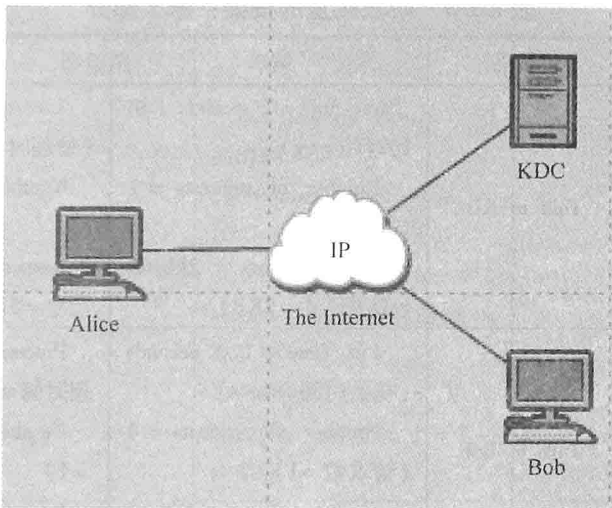


图 L5.2 这项作业的网络拓扑

3）验证链路连通性（见 2.6.2 节）。

4）将 The Internet 云丢弃所有到达的报文的 0.0%，并为通过它的所有流量引入额外的 100ms 的延迟（见实验室作业 1 了解配置细节）。

接下来，将为 Secure AB Talk 和 AB Talk 应用的建模配置认证、加密的传输和纯文本传输任务。

1）添加 *Task Config* 对象，并将之命名为 Tasks（见 6.2.1 节）。

2）依据如下规则（也在表 L5.2 和表 L5.3 中做了概述）：

表 L5.2 阶段配置的摘要：第 1 部分

任务名	阶段名	在... 之后开始	源	目的地
Authentication	Talk to KDC	Application Starts	Alice	KDC
	Talk to Bob	Previous Phase Ends	Alice	Bob
	Verify Bob	Previous Phase Ends	Alice	None
Secure Trans-mission	Alice to Bob	Application Starts	Alice	Bob
	Bob to Alice	Application Starts	Bob	Alice
Plain Text Tran-smission	Alice to Bob	Application Starts	Alice	Bob
	Bob to Alice	Application Starts	Bob	Alice

① Authentication 任务

阶段 Talk to KDC 对认证过程的步骤 1~3 建模。这个阶段与应用开始的时间相同，其工作过程如下：

表 L5.3　阶段配置的摘要：第 2 部分

任务名	阶段名	请求	响应
Authentication	Talk to KDC	Init time = 0 seconds（初始时间 = 0s） Number of requests = 1（请求数） Request size = 2kbytes（请求尺寸 = 2KB）	Processing time ≅ 0.5 seconds（处理时间≅0.5s） Number of replies（应答数 = 1） Response size = 5kbytes（响应尺寸 = 5KB）
	Talk to Bob	Init time ≅ 0.4 seconds（初始时间≅0.4s） Number of requests = 1（请求数 = 1） Request size = 4kbytes（请求尺寸 = 4KB）	Processing time ≅ 1 second（处理时间≅1s） Number of replies = 1（应答数 = 1） Response size = 1kbytes（响应尺寸 = 1KB）
	Verify Bob	Init time ≅ 0.2 seconds（初始时间≅0.2s） Num requests = 0（请求数 = 0）	N/A
Secure Transmission	Alice to Bob	Init time ≅ 0.2 seconds（初始时间≅0.2s） Number of requests = 1000（请求数 = 1000） Inter - request time ≅ 1.2 second（请求间隔时间≅1.2s） Request size = 1 - 10kbytes（请求尺寸 = 1 ~ 10KB）	N/A
	Bob to Alice	Init time ≅ 0.2 seconds（初始时间≅0.2s） Number of requests = 1000（请求数 = 1000） Inter - request time ≅ 1.2 second（间隔请求时间≅1.2s） Request size = 1 ~ 10kbytes（请求尺寸 = 1 ~ 10KB）	N/A

（续）

任务名	阶段名	请求	响应
Plain Text Transmission	Alice to Bob	Init time = 0 seconds（初始时间 = 0s） Number of requests = 1000（请求数 = 1000） Inter - request time ≅ 1 second（请求间隔时间 ≅ 1s） Request size = 1 ¬ 10kbytes（请求尺寸 = 1 ~ 10KB）	N/A
	Bob to Alice	Init time = 0（初始时间 = 0） Number of requests = 1000（请求数 = 1000） Inter - request time ≅ 1 second（请求间隔时间 ≅ 1s） Request size = 1 - 10kbytes（请求尺寸 = 1 ~ 10KB）	N/A

- 尺寸为 2KB 的单个消息由 Alice 发送到 KDC。Alice 不为产生这条消息花费任何初始化时间。
- 在向 Alice 发回尺寸为 5KB 的单条响应消息之前，KDC 服务器花费大约 2s 处理 Alice 的请求。

阶段 Talk to Bob 对认证过程的步骤 4 ~ 8 建模。这个阶段在 Talk to KDC 阶段完成之后开始，其工作过程如下：

- 在向 Bob 发送尺寸为 4KB 的一条消息之前，Alice 花费大约 0.9s 来处理 KDC 的响应。
- 在将尺寸为 1KB 的一条响应消息发回 Alice 之前，Bob 花费大约 1s 来处理 Alice 的请求。

阶段 Verify Bob 对认证过程的最后一步进行建模，其中 Alice 花费 0.2s 处理 Bob 的响应消息。在 Talk to Bob 阶段完成之后，这个阶段开始。

② Secure Transmission 任务

阶段 Alice to Bob 对从 Alice 到 Bob 的安全通信进行建模，该通信由加密消息，以及之后从 Alice 将它们发送 Bob 这两步组成。假定被加密消息的尺寸是均匀分布在 1KB 和 10KB 之间的，且花费大约 0.2s 加密一条消息（即在发送一条消息之前，Alice 花费

0.2s 对消息进行加密）。大约每隔 1s Alice 产生一次到 Bob 的一条新消息。由 Alice 发送的两条连续请求之间的总时间由消息间的产生时间（1s）和花费在加密消息的时间（0.2s）组成。假定，在 Alice 与 Bob 的通信过程中她产生大约 1000 条消息，且每条消息可装入到单条应用报文。因为将从 Bob 到 Alice 的通信建模为一个独立的阶段，所以在这个阶段过程中不产生响应消息。将这个阶段配置为在前一任务完成之后开始（即应该将属性 **Start Phase After** 设置为值 Application Starts）。

阶段 Bob to Alice 对从 Bob 到 Alice 的安全通信进行建模，由加密消息和之后将消息从 Bob 发送到 Alice 组成。这个阶段的配置与 Alice to Bob 阶段的配置相同，例外是这个阶段的源节点和目的地节点角色做了互换。阶段 Bob to Alice 在与 Alice to Bob 阶段的同一时间开始。

③ Plain Text Transmission 任务

该任务也由两个阶段组成：Alice to Bob 和 Bob to Alice。这些阶段的配置与 Secure Transmission 任务的那些阶段配置相同，例外是，报文不被加密并以纯文本发送。结果是，在没有任何附加加密延迟的条件下，两个阶段可发送数据（即初始时间应该设置为 0，且请求间隔时间应该设置为 1s）。

当可应用时（如将一个属性设置为一个非零值），使用指数分布指定初始化时间、请求间隔时间和报文间隔时间。在 Authentication 阶段使用常数分布指定请求消息和响应消息的尺寸，并在传输阶段过程中使用最小输出为 1KB 和最大输出为 10KB 的均匀分布指定请求尺寸。使用常数分布指定请求消息数。

一旦指定了所有的任务，将定义两个定制应用：Secure AB Talk 和 AB Talk（见 6.3 节）。Secure AB Talk 定制应用由两个任务组成，Authentication 和 Secure Transmission 以串行顺序执行，而 AB Talk 定制应用则由单个任务组成：Plain Text Transmission。

接下来，将定义两个概要 Secure AB Talk User 和 AB Talk User，分别支持 Secure AB Talk 和 AB Talk 定制应用。将每个概要配置为仅在仿真过程中运行一次应用。在网络中部署定义的用户概要（见 7.2 节和 7.4 节）。

最后，为 Custom Application 配置仿真收集所有全局统计量（见 4.2.4 节和 5.6 节），之后将仿真执行 2000s（见 4.3 节）。

问题

Q1. 定制应用的每个阶段的响应时间为多少？

Q2. 如何相互比较 Secure Transmission 和 Plain Text Transmission 任务的响应时间？这些结果与个体阶段所报告的响应时间一致吗？

Q3. 如何比较 Secure AB Talk 和 AB Talk 应用的响应时间？哪个应用完成需要更多时间？为什么？

Q4. Authentication 任务的响应时间为多少？Authentication 任务如何影响整体应用性能？

L5.3 应用性能与较短的应用时间

在作业的这部分，将详细研究应用时长如何影响整体性能。

1）复制当前场景，将新的场景命名为AB_ Talk_ Short（见1.7节）。

2）将在Secure Transmission和Plain Text Transmission任务中的数据传输阶段的定义，改变为每阶段仅产生5条消息，而不是1000条（见6.2.3节）。

3）更新Secure AB Talk User和AB Talk User概要，使之运行其响应的应用15s，并具有无限的可重复性（见7.2节）。

4）执行仿真并详细研究所收集的结果。

问题

Q5. 定制应用的每个阶段和每项应用的平均响应时间是多少？

Q6. Secure AB Talk和AB Talk应用的平均响应时间比较的情况如何？哪个应用的完成要求更多时间？为什么？

Q7. Authentication任务的平均响应时间是多少？Authentication任务如何影响整体应用性能？在authentication上花费的Secure AB Talk应用响应时间占百分之几？

L5.4 应用性能及通过多条报文发送应用消息

在这部分作业中，将详细研究当单条应用消息分成多条报文（每条报文都要加密）时会发生什么情况。实施如下步骤仿真这样一个场景：

1）复制场景AB_ Talk，命名新场景为AB_ Talk_ Packets。

2）在Secure Transmission任务中，如下改变数据传输配置（即为两个阶段改变属性**Source ->Dest Traffic**的值）：

①设置初始化时间为0s（常数分布），原因是应用不加密消息；相反，它加密消息内的个体报文。

②设置请求间隔时间为1s。

③设置每个请求的报文数为5（常数分布）。

④设置报文间隔请求时间为0.21s（指数分布）：0.2s加密一条报文，0.01s用来创建个体报文，方法是对原消息分片。

3）如下改变Plain Text Transmission任务中的数据传输配置：

①设置每个请求的报文数为5（常数分布）。

②设置报文请求间隔时间为0.01s（指数分布）：不需要加密；由于消息分片，产生报文间隔时延。

4）执行仿真并详细研究所收集的结果。

问题

Q8. 定制应用的每个阶段和每项任务的响应时间是多少？

Q9. 画出一个表，比较在 AB_ Talk 和 AB_ Talk_ Packets 场景中收集的数据传输阶段的响应时间。您观察到哪些差异？为什么？

Q10. 在 AB_ Talk 和 AB_ Talk_ Packets 场景中收集的应用响应时间比较情况如何？为什么？

实验室作业6：最大传输单元对应用性能的影响

针对这些实验室作业，建议阅读第1～5章和第7章。

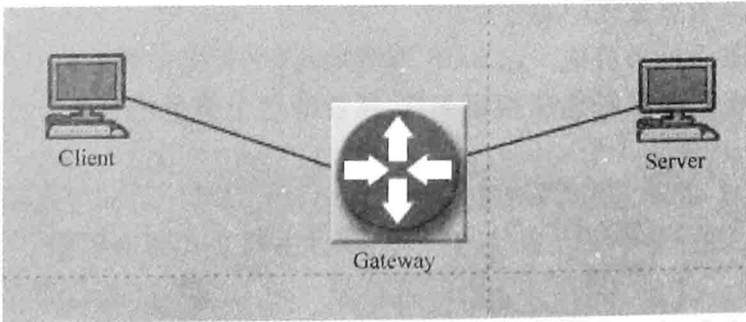
L6.1 引言

在这项实验室作业中，将研究最大传输单元（MTU）尺寸如何影响应用性能。具体而言，将详细研究在网络层中变化的MTU值如何改变一个FTP应用的响应时间。回忆一下，MTU尺寸确定应用数据如何由IP分片。特别地，分片尺寸通过如下两个因素影响应用响应时间：

1）如果应用数据被分成许多小的分片，那么通过网络承载的总流量要大于当数据被分成少量大型分片的总流量。流量的增加是由于个体分片携带的附加首部信息造成的（即每个IP分片携带网络和数据链路层首部）。当网络层携带的总流量增加时，应用响应时间增加（即传输同样的应用数据量，要花费较长时间）。因此，当应用数据被分片成许多小的片时，应用响应时间会增加。

2）另外，传输一个大型分片的整体端到端时延，大于传输一个小型分片序列的端到端时延，原因是多跳上的“流水线”效应造成的。在传输一个大型分片所花费的时间中，几个小的分片可由源传输，由下一跳接收，并进一步转发到网络中。将数据分片成较小的数据块的做法支持“流水线”，这降低了一个应用所经历的响应时间。因此，当应用数据被分片成许多小片时，应用响应时间会减少。

明显的是，这两个因素具有相反的效应，且应该存在某个“最优”MTU尺寸，该尺寸将最小化应用响应时间。在这项实验室作业中，将为计算最优MTU尺寸推导一个公式，并将计算与一个仿真研究的结果做比较。考虑如图L6.1所示的网络拓扑，其中客户端节点每隔10s上传50 000字节的文档。在这项作业中，将在客户端节点和服务器节点网络中改变MTU尺寸，从500字节到3000字节，以500字节为增量增加，之后将详细研究MTU尺寸如何影响FTP上传响应时间。



图L6.1 Two_ Hop_ network 场景的网络拓扑

L6.2 创建仿真模型

在这部分作业中，将为这项实验室作业创建网络的一个仿真模型。

1）创建一个新项目和一个空场景，分别命名为 Assignment 06 和 Two_ Hop_ Network（见 1.6.2 节）。

2）使用表 L6.1 中指定的节点和链路模型，创建如图 L6.1 所示的网络拓扑（见 2.3 节和 2.4 节）。

表 L6.1 网络拓扑的节点模型和链路模型

对象名	对象模型
Client Server	*ppp_ wkstn* 节点对象
Gateway	*ethernet4_ slip8_ gtwy* 节点对象
Client < - > Gateway Server < - > Gateway	*PPP_ DS1_ int* 双工链路对象

3）验证链路连通性（见 2.6.2 节）。

4）添加 *Application Config* 和 *Profile Config* 节点对象（见 5.3.1 节和 7.2.1 节）。

5）配置一个 FTP 应用，使之每隔 10s 产生 50 000 字节的文件上传（见 5.4.3 节）。

6）配置一个 FTP 用户概要，对于仿真时长期间，运行一次所定义的 FTP 应用（见 7.2 节）。

7）在网络中部署所定义的 FTP 用户概要，使 Client 和 Server 节点分别作为 FTP 应用的源和服务器（见 7.4 节）。

接下来将配置要在这个仿真中收集的统计量，为 Client 和 Server 节点指定 MTU 值（即提升 MTU 属性），并执行仿真。

8）配置仿真收集 FTP 应用的所有全局统计量（见 4.2.4 节和 5.6 节）。

9）打开 Client 节点 **Edit Attributes** 菜单，右击属性 IP... IP Host Parameters... Interface Information... MTU，并将那个属性值设置为 promoted（见 3.5.1 节）。

10）为 Server 节点重复相同过程。

11）指定所提升的数值值，以 500B 为增量从 500B 变化到 3000B。将需要在 Client 和 Server 节点上使用通配符选项指定 MTU 属性的这个值范围（见 3.5 节、4.3.5 节和 4.3.6 节）。

12）执行仿真 400s（见 4.3 节）。

问题

Q1. 对于每个 MTU 值，FTP 上传响应时间是多少？

Q2. 哪个 MTU 值得到最小的 FTP 上传响应时间？为什么？（提示：像在 L6.1 节讨

论的那样，考虑分片如何影响应用响应时间）。

Q3. 假定数据链路层首部尺寸为7字节，IP首部尺寸为20字节。同样假定$T_{transmission}$（传输D字节的应用数据所需时间）可依据式（L6.1）如下近似。为了简化计算，忽略FTP和TCP首部的尺寸（相比于D字节的应用数据而言是较小的），并将发送数据所需的报文数近似为数据尺寸（D字节）和MTU尺寸的比值。

$$T_{transmission} = \left[\frac{D}{\mathrm{MTU}}\right] \times (H + \mathrm{MTU}) \div R \approx \frac{D(H + \mathrm{MTU})}{\mathrm{MTU} * R} \quad (\mathrm{L6.1})$$

其中D是应用数据量；MTU是最大传输单元尺寸；H是数据链路层和IP首部尺寸之和（27字节）；R是传输速率。

假定传播和处理时延是可忽略的。使用式（L6.1），为端到端时延推导一个公式，是从源节点传输数据第一个比特的时间直到到达目的地的数据最后一个比特的时间计算得到的。假定数据是通过一个报文交换网络进行传输的，在源和目的地之间有N跳，其中一跳对应于穿越一条链路。（提示：数据D到达目的所用总时间等于源节点在第一跳传输数据D需要的时间架上最后一条数据报文穿越路径上的剩余$N-1$跳到达目的地所用的时间。可假定所有报文具有相同尺寸）。

Q4. 由为Q3推导的公式，计算最优MTU尺寸。（提示：为计算最优MTU尺寸，对Q3计算的方程求MTU的导数，并将结果等于0。对MTU求解得到的方程）。

Q5. 使用推导得到的方程，计算如下情况下的最优MTU尺寸，即在具有两跳的一个网络上（即$N=2$），从源到目的地传输50 000B应用数据，这是在仿真场景中所使用网络的情形。假定网络和数据链路层首部的总尺寸为27B。将得到的最优MTU值与仿真产生的结果做比较。仿真结果与您的计算一致吗？为什么一致或为什么不一致？

L6.3 增加网络中的跳数

现在将重复试验，方法是将网络中的跳数增加到4，并详细检查应用响应时间和MTU尺寸之间的相互作用如何受到所增加跳数的影响。

1）复制当前场景，将新场景命名为Four_ Hop_ Network（见1.7节）。

2）在Client和Server之间添加另外两个网关节点。

3）通过DS1链路连接新节点，形成如图L6.2所示的拓扑。

4）执行仿真，并详细检查得到的结果。

问题

Q6. 对于新场景中的每个MTU值，FTP上传响应时间是多少？

Q7. 哪个MTU值得到最小的FTP上传响应时间？这些结果与L6.2节中收集的数据比较，情况如何？为什么？

Q8. 使用L6.2节中推导得到的最优MTU尺寸公式，计算针对这个仿真场景应该得到最小FTP上传响应时间的MTU值。注意现在跳数是4。与仿真结果比较，计算得到

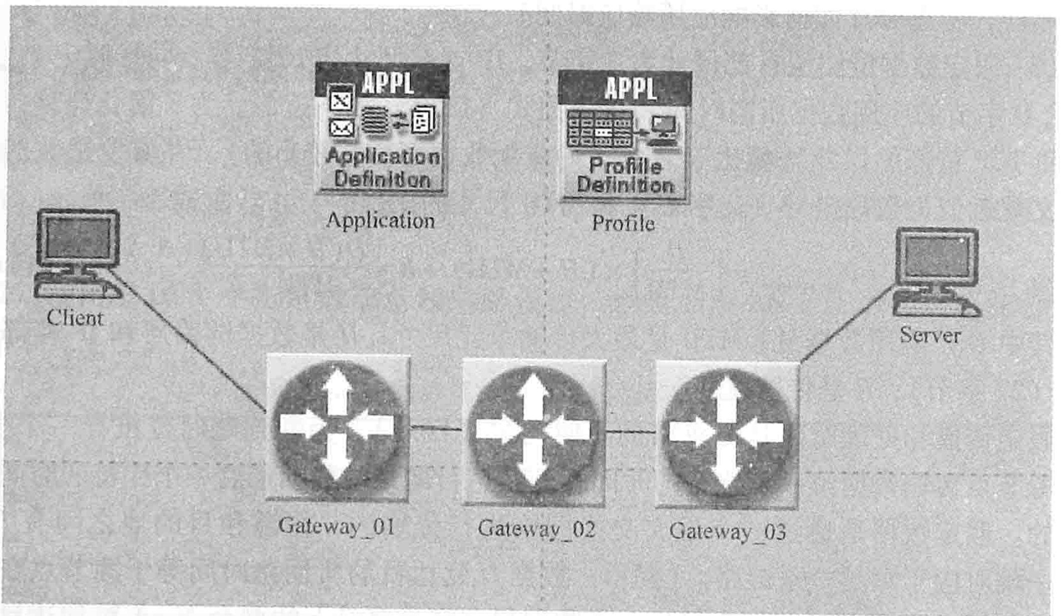


图 L6.2 Four_ Hop_ Network 场景的网络拓扑

的值如何？仿真结果支持计算得到的最优 MTU 值吗？

实验室作业7：传输协议——TCP和UDP

针对这项实验室作业，建议阅读第1～5章、第7章和第8章。

L7.1 引言

在这项实验室作业中，将比较运行在不同传输协议上的应用的性能。特别地，将详细研究应用性能如何受到TCP和UDP的影响。此外，在创建复杂OPNET仿真模型、建立标准应用和开发用户概要过程中将精练（hone）您的技巧。

考虑如下情况：Tim“万维网迷”（the Web Dude）开发了一个超级秘密（简称“James Bond”（吉姆斯．邦德））间谍应用，在甚至没有任何人知道一些秘密的信息正在被传输的情况下，用于在网络上安全地加密和传输信息。该应用将信息编码为一系列图像，并称为Hermes（赫尔墨斯），是以希腊神话中众神的使者命名的。对于监测由Hermes所产生的流量的一名外行来说，将看起来是大量小图像正在传递，而实际上这些图像携带一些秘密信息。

Hermes可部署在具有各种丢失特征的环境中。Tim的雇主确定了Hermes将工作于其中的如下环境类型：

1）0.00%丢失（在一个无丢失的LAN中传递）。

2）0.00%丢失（在没有数据丢失的互联网上传递）。

3）0.5%丢失（在具有少量数据丢失的互联网上传递）。

4）1.00%丢失（在具有一些数据丢失的互联网上传递）。

5）5.00%丢失（在具有大量数据丢失的互联网上传递）。

Hermes在尺寸为10KB的10个连续图像（每隔100ms产生一个）中编码单个秘密信息块。为了使Hermes正常工作，所有的10个图像应该在5s内接收到。

Tim“万维网迷”雇佣您作为咨询人员，确定Hermes要在上述环境内使用的最适合的传输协议。不幸的是，Tim不能泄露Hermes实现的细节，因为它是高度秘密。但是，Tim为您提供了足够的细节，来创建一个仿真模型：

1）Hermes是一个FTP类的客户端—服务器应用。

2）客户端仅产生数据下载的请求。

3）客户端每隔100ms产生一次请求。

4）被请求的文件尺寸是10 000B。

5）客户端如下运行Hermes应用：

① 客户端产生10个文件的请求集合。

② 在产生另一个请求集合之前，客户端等待9s。

③ 客户端在100s内重复步骤1和步骤2，那么暂停100s，并再次启动。

④ 如有必要，客户端可以需要的次数重复步骤1~步骤3。

L7.2 在没有数据丢失的环境中的Hermes应用

在这部分作业，将创建一个网络环境（Hermes应用工作于其中）的一个仿真模型。

1）创建一个新项目和一个空场景，分别命名为Assignment 07和Hermes with No Loss（见1.6.2节）。

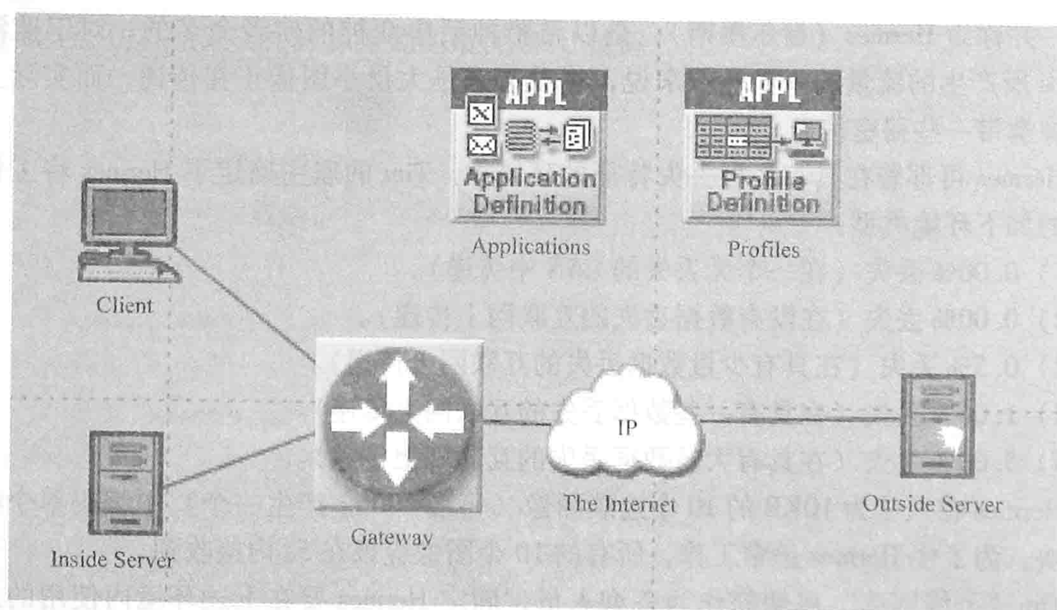
2）使用表L7.1中指定的节点和链路模型，创建如图L7.1所示的网络拓扑（见2.3节和2.4节）。

3）验证链路连通性（见2.6.2节）。

4）添加*Application Config*和*Profile Config*节点对象。

5）创建两个等同的FTP应用，称之为Hermes Local和Hermes over the Internet，每隔100ms产生10 000B的文件下载（见5.4.3节）。

6）配置一个概要，称为Local Use，如下利用Hermes Local应用（见7.2节）。



图L7.1 网络拓扑

① 在概要内指定应用特征：

- 将应用启动时间偏移设置为0s，指明应用在概要的相同时间启动。
- 因为Hermes应用每隔100ms产生一个请求，所以将应用时长设置为1s来仿真请求10个文件。
- 为了仿真Hermes应用在发送下一批次请求（即步骤2）之前等待9s的情况，将应用重复间隔时间设置为9s，重复次数设置为Unlimited，重复模式设置为Serial。

② 为了仿真Hermes应用在重复步骤1和2之前暂停100s的情况，如下配置概要

特征。

- 设置概要工作模式为Serial。
- 设置概要启动时间为100s。
- 设置概要时长为100s。
- 设置概要可重复性为Unlimited。

③ 这样一种配置将使概要运行100s，之后等待100s，之后再次启动，重复直到仿真结束。为简化数据分析，当设置时间偏移和其他概要或应用特征时，仅使用常数分布。

7）复制概要Local Use，并如下改变其配置：

① 将新概要的名字改变为Use over the Internet。

② 改变新概要所采用的应用为Hermes over the Internet。

③ 保持其他概要属性不变。

8）在网络中部署所定义的概要，从而使Client作为Hermes应用的一个源，而节点Inside Server和Outside Server分别是Local Use和Use over the Internet概要中Hermes应用的FTP服务器（见7.4节）。

表 L7.1　网络拓扑摘要

对象名	对象模型
Client	*ethernet_ wkstn_ adv* 节点对象
Inside Server	*ethernet_ server_ adv* 节点对象
Outside Server	ppp_ server_ adv 节点对象
Gateway	*ethernet4_ slip8_ gtwy* 节点对象
Client < - > Gateway Inside Server < - > Gateway	100*BaseT* 链路对象
Gateway < - > The Internet The Internet < - > Outside Server	*PPP_ DS1_ int* 链路对象

接下来，将配置穿越互联网的报文所经历的延迟，配置要在这个仿真中收集的统计量，为Client、Inside Server和Outside Server节点指定传输协议，并执行仿真。

1）设置The Internet节点中的报文延迟为100ms。

需要设置节点的属性 **Performance Metrics... Packet Latency**（**secs**）的值为0.1。

2）配置仿真收集如下统计量（见4.2.3节和5.6节）：

① 所有 **Client FTP** 节点统计量。

② 所有 *point - to - point* 链路统计量。

3）通过设置节点属性 **Applications: Transport Protocol Specification... FTP Transport** 的值为Promoted（见3.5.1节），配置端节点Client、Inside Server和Outside Server，使之运行在不同传输协议上。

4）指定被提升属性的值为TCP和UDP。可能需要使用通配符，在所有端节点上指定 **FTP Transport** 的值（见3.5节、4.3.5节和4.3.6节）。

5）执行仿真3000s。

问题

一旦仿真完成，详细研究TCP和UDP如何影响应用响应时间和资源消耗。应该在如下四种情形中详细检查Hermes应用的下载响应时间以及连接到服务器链路上的利用率和吞吐量：

1）LAN中TCP上的Hermes。

2）LAN中UDP上的Hermes。

3）穿越互联网在TCP上传输的Hermes，没有数据丢失。

4）穿越互联网在UDP上传输的Hermes，没有数据丢失。

Q1. 为上述每种情形，创建一个表，表中包含Hermes应用的最大、最小和平均下载响应时间。基于对TCP、UDP的理解和网络的物理配置，解释观察到的结果。

1）当仅在LAN中使用时，TCP或UDP中的哪个协议得到Hermes应用的较短响应时间？为什么？

2）当在互联网上传输数据时，TCP或UDP中的哪个协议得到Hermes应用的较短响应时间？为什么？

3）当运行在UDP上时，LAN或互联网中的哪种网络环境得到Hermes应用的较短平均下载响应时间？为什么？

4）当运行在TCP上时，LAN或互联网中的哪种网络环境得到Hermes应用的较短平均下载响应时间？为什么？

Q2. 针对四种情形中的每种情形，详细研究连接到服务器节点的链路上的利用率和吞吐量。基于对TCP、UDP的理解和网络的物理配置，解释观察到的结果。

1）当仅在LAN中使用时，TCP或UDP中的哪个协议得到Hermes应用的较低的链路利用率和吞吐量？为什么？

2）当在互联网上传输数据时，TCP或UDP中的哪个协议得到Hermes应用的较低的链路利用率和吞吐量？为什么？

3）当运行在UDP上时，LAN或互联网中的哪种网络环境得到Hermes应用的较低的链路利用率和吞吐量？为什么？

4）当运行在TCP上时，LAN或互联网中的哪种网络环境得到Hermes应用的较低的链路利用率和吞吐量？为什么？

Q3. 基于仿真结果，当在没有报文丢失的一个网络环境中使用Hermes时，建议Tim使用哪种传输协议？为什么？

L7.3 有数据丢失环境中的Hermes应用

在真实生活中，互联网很少提供无丢失的数据交付。在这部分作业中，将详细研究

以某个概率丢失数据的互联网环境中，Hermes应用的响应时间和资源消耗情况。

1）复制当前场景，命名新的场景为 Hermes with Loss（见1.7节）。

2）提升 The Internet 节点中的属性 **Performance Metrics... Packet Discard Ratio**。

3）将提升属性 **Packet Discard Ratio** 的值设置为0.5%、1.0%和5.0%。

4）执行仿真并详细研究所收集的结果。

问题

当详细研究仿真结果时，可忽略如下情形下收集的数据，其中Hermes应用通过局域网发送流量。相反，要详细检查如下情形下的应用响应时间，以及 The Internet 和 Outside Server 节点之间链路上的利用率和吞吐量，其中Hermes通过互联网发送流量。出于本节中所有问题的目的，假定，在偶然的报文丢失情况下，Hermes可恢复编码的信息。通常情况下，为从单条报文丢失中恢复，Hermes需要额外的0.5s处理时延完成其数据恢复计算。

Q4. 显示图形，其中展示了运行在UDP之上通过具有0.5%、1.0%和5.0%报文丢失的一个网络传输的Hermes应用的应用响应时间。哪个报文丢失值给出最短响应时间？为什么？考虑在报文丢失的情形中与数据重建相关联的额外处理时延，**Packet Discard Ratio** 属性的哪个值得到最短应用响应时间？

Q5. 相比于没有丢失的情况，运行在UDP之上在有丢失的网络环境中Hermes应用的响应时间如何变化？为什么？

Q6. 显示图形，其中展示了运行在TCP之上通过具有0.5%、1.0%和5.0%报文丢失的网络传输的Hermes应用的应用响应时间。考虑在报文丢失的情形中与数据重建相关联的额外处理时延，**Packet Discard Ratio** 属性的哪个值得到最短应用响应时间？相比于没有丢失的情况，运行在TCP之上在有丢失的网络环境中Hermes应用的响应时间如何变化？为什么？

Q7. 相比于Hermes运行在UDP之上无丢失的情况，在有丢失的网络环境中链路利用率和吞吐量如何变化？为什么？

Q8. 相比于Hermes运行在TCP之上无丢失的情况，在有丢失的网络环境中链路利用率和吞吐量如何变化？为什么？

Q9. 基于这些观察结果，对于经历数据丢失环境中的Hermes应用，建议Tim使用哪个传输协议？为什么？

实验室作业 8：TCP 功能

对于这项实验室作业，建议阅读第 1 ~5 章、第 7 章和第 8 章。

L8.1 引言

在这部分作业中，将探究精细调整 TCP 配置的技术。特别地，将探究诸如 Nagle 算法、最大分段尺寸（MSS）、接收方窗口尺寸和各种 TCP 风格等 TCP 功能如何影响应用性能。

L8.2 Nagle 算法

Nagle 算法解决小型报文尺寸的问题。与直接发送小型数据报文（每条报文将携带 40 字节的 TCP 和 IP 首部信息）的做法不同，Nagle 算法建议缓冲数据，直到应用提供更多的数据进行发送或直到其所有外发报文都被确认。在这部分作业中，将详细研究 Nagle 算法如何影响应用时延和在线路上发送的流量总量。

1）创建一个新的项目和一个空场景，分别命名为 Assignment 08 和 Nagles Algorithm（见 1.6.2 节）。

2）使用表 L8.1 中指定的节点和链路模型，创建如图 L8.1 所示的网络拓扑（见 2.3 节和 2.4 节）。

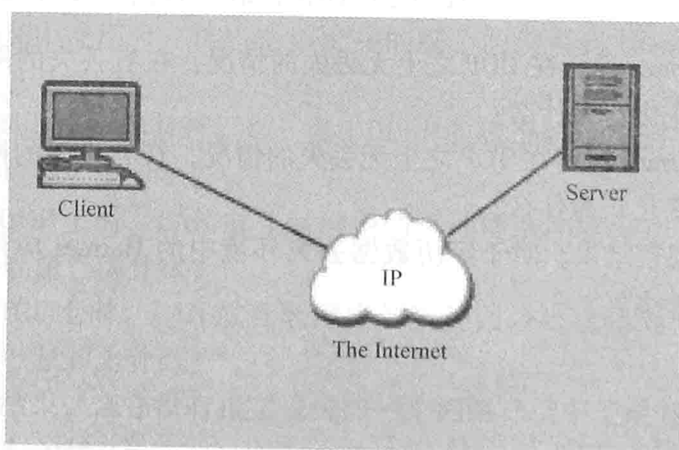


图 L8.1　网络拓扑

3）验证链路连通性（见 2.6.2 节）。

4）添加 *Application Config* 和 *Profile Config* 节点对象。

表 L8.1 网络拓扑摘要

对象名	对象模型
Client	*ppp_ wkstn* 节点对象
Server	*ppp_ server* 节点对象
The Internet	*ip32_ cloud* 节点对象
网络中的所有链路	*PPP_* 33*K* 链路对象

5）创建一个远程登录应用，称为 Telnet，该应用产生进出终端的 1 字节命令。为简化数据分析，使用一个常数分布指定命令的尺寸。通过单击按钮 **Promote**，提升属性 **Inter – Command Time (seconds)**。欲了解有关远程登录应用的更多信息，参见 5.4.6 节。

6）配置一个概要，称之为 Telnet User，它以默认设置运行 Telnet 应用（见 7.2 节）。

7）将所定义的概要部署在网络中，使 Client 作为源，Server 是 Telnet 应用的一个远程登录服务器（7.4 节）。

8）如下指定节点配置：

① 通过指定属性 **Performance Metrics... Packet Latency (secs)** 的值，将 The Internet 节点中的报文延迟设置为 100ms。

② 提升 Client 和 Server 节点中的属性 **TCP... TCP Parameters... Nagle Algorithm**。

9）配置仿真收集如下统计量：

① **Client Remote Login** 节点统计类中的 **Response Time (sec)**。

② 所有 point – to – point 链路统计量。

10）如下指定所提升属性的值：

① 设置 **TCP Parameters... Nagle Algorithm** 属性的值为 Disabled 和 Enabled。

② 设置属性 **Inter – Command Time** 的值为 exponential (0.1) 和 exponential (10)。

11）运行仿真 1h。

问题

对于如下四种情况，详细研究应用响应时间并显示一个图形：

1）Nagle 算法被禁止，且命令间隔时间设置为 0.1s。

2）Nagle 算法被激活，且命令间隔时间设置为 0.1s。

3）Nagle 算法被禁止，且命令间隔时间设置为 10s。

4）Nagle 算法被激活，且命令间隔时间设置为 10s。

Q1. 当 Nagle 算法被激活和禁止时，应用响应时间如何变化？为什么？

Q2. 上述四种场景中，哪些场景得到最低的应用响应时间，哪些场景得到最高的应用响应时间？为什么？

详细研究 Client 和 The Internet 之间链路上的吞吐量，显示一个图形，其中比较上

述四种场景中的链路吞吐量，并回答如下问题：

Q3. 当 Nagle 算法被激活和禁止时，链路吞吐量如何变化？为什么？

Q4. 上述四种场景中，哪些场景得到 Client 和 The Internet 之间链路上最低吞吐量和最高吞吐量？为什么？

L8.3 端到端时延对 Nagle 算法的影响

在这部分作业中，将详细研究互联网中报文延迟如何影响 Nagle 算法性能。

1）复制当前场景，并命名新场景为 Nagles Algorithm with Delay。

2）提升 The Internet 节点中的属性 **Performance Metrics... Packet Latency** （**secs**）。

3）如下指定被提升属性的值：

① 设置 **TCP... TCP Parameters... Nagle Algorithm** 属性的值为 Disabled 和 Enabled。

② 设置属性 **Inter – Command Time** 的值为 exponential（0.1）。

③ 设置属性 **Performance Metrics... Packet Latency**（**secs**）的值为 0.1s 和 0.001s。

4）运行仿真 1h。

问题

详细研究在仿真过程中收集的应用响应时间和链路吞吐量等统计量。显示与作业前一部分中相同的图形集，并回答如下问题：

Q5. 当 Nagle 算法被激活和禁止时，报文延迟如何影响应用响应时间？为什么？

Q6. 当报文延迟的值改变时，应用响应时间如何改变？您认为所报告的结果与场景的配置一致吗？为什么一致或为什么不一致？

Q7. 当 Nagle 算法被激活和禁止时，报文延迟如何影响链路吞吐量？为什么？

Q8. 哪个报文延迟的值得到链路上的最低吞吐量？为什么？

L8.4 TCP 的 MSS 尺寸对应用性能的影响

在这部分作业中，将详细研究 MSS 对应用性能的影响。

1）创建一个新场景，命名为 TCP MSS。

2）使用表 L8.1 中指定的节点模型，创建如图 L8.1 所示的网络拓扑。但是，使用 PPP_ DS1 链路模型连接网络中的各节点。

3）验证链路连通性。

4）添加 *Application Config* 和 *Profile Config* 节点对象。

5）创建一个 FTP 应用，称之为 FTP，它每隔 3min 发出一条 get 命令，下载一个 1Mb 大小的文件。使用一个常数分布指定文件尺寸和请求间隔时间（见 5.4.3 节）。

6）配置一个概要，称之为 FTP User，它以默认设置运行 FTP 应用（见 7.2 节）。

7）在网络中部署所定义的概要，从而使 Client 作为 FTP 应用的源，节点 Server 作为一个服务器（见 7.4 节）。

8）如下指定节点配置：

① 设置 The Internet 节点中的报文延迟为 100ms。

② 提升 Client 和 Server 节点中的属性 **TCP... TCP Parameters... Maximum Segment Size（bytes）**。

③ 确保两个端点处接收缓冲尺寸设置为默认值 8760B。

9）配置仿真收集如下统计量：

① **FTP** 节点统计类中的 **Download Response Time（sec）**。

② **TCP** 节点统计类中的 **Segment Delay（sec）**。

③ 所有 *point - to - point* 链路统计量。

10）设置被提升属性 **TCP Parameters... Maximum Segment Size（bytes）** 为如下值：Auto - Assigned、500、1000、1400、1500、2000 和 5000。

11）运行仿真 10min。

问题

针对下载响应时间、TCP 分段时延以及 Client 和 The Internet 之间链路上的点到点吞吐量，详细研究仿真结果。

Q9. 显示图形，其中展示下载响应时间和 MSS。哪个 MSS 值得到最大下载响应时间？为什么？哪个 MSS 值得到最大下载响应时间？为什么？1500 字节 MSS 和 1400 字节 MSS 这两者的下载响应时间相比较，情况如何？为所观察到的行为，给出一个解释。

Q10. 显示图形，展示 TCP 分段时延和 MSS。哪个 MSS 值得到最低 TCP 分段延迟？为什么？哪个值得到最大 TCP 分段时延？为什么？

Q11. 显示图形，展示 Client 和 The Internet 之间链路上的 TCP 吞吐量和 MSS。哪个 MSS 值得到最低的链路吞吐量？为什么？哪个值得到最大的链路吞吐量？为什么？

L8.5 TCP 的接收缓冲尺寸对应用性能的影响

在这部分作业中，将详细研究接收缓冲尺寸对应用性能的影响。回忆一下，在 OPNET 中，属性 **Receive Buffer（bytes）** 指定在接收方处可用的空间总量，在到达的数据被转发到高层之前，该空间用来存储到达的数据。这个值不同于通告的接收窗口（**rwnd**），这是在接收缓冲中当前可用来存储到达数据的空闲空间量。

1）复制 TCP MSS 场景，并命名新场景为 TCP Receive Buffer。

2）如下修改节点配置：

① 在 Client 和 Server 节点处设置属性 **TCP Parameters... Maximum Segment Size（bytes）** 为 Auto - Assigned。

② 提升属性 **TCP Parameters... Receive Buffer**（**bytes**）。

3）如下修改全局属性的配置：

① 从 **Object Attributes** 表中删除以前提升的属性 **TCP Parameters... Maximum Segment Size**（**bytes**）。

② 添加被提升属性 **Receive Buffer**（**bytes**），之后将之设置为如下值：8760、32768、65535 和 Default。

4）运行仿真 10min。

问题

针对下载响应时间以及 Client 和 The Internet 之间链路上的点到点吞吐量，详细研究仿真结果：

Q12. 显示图形，展示下载响应时间和接收缓冲尺寸。哪个接收缓冲尺寸值得到最低下载响应时间？为什么？哪个接收缓冲尺寸值得到最大下载响应时间？为什么？

Q13. 基于所观察到的结果和如下事实，即默认情况下，OPNET 将接收缓冲尺寸设置为四倍的 MSS 值，猜测接收缓冲尺寸的默认值。

Q14. 显示图形，展示 Client 和 The Internet 之间链路上的 TCP 吞吐量和接收缓冲尺寸。哪个接收缓冲尺寸值得到最低的链路吞吐量？为什么？哪个值得到最大的链路吞吐量？为什么？您认为所报告的链路吞吐量结果与您对默认接收缓冲尺寸的猜测一致吗？为什么一致或为什么不一致？

L8.6 TCP 拥塞控制

在这部分作业中，将详细研究如下情况：存在报文丢失时，各种 TCP 风格的行为如何。具体而言，将详细研究 Tahoe、Reno、New Reno 和 SACK TCP 等风格的性能。为实施这项研究，需要如下创建一个新的 OPNET 场景：

1）创建一个新的场景，并命名为 TCP Congestion Control。

2）使用表 L8.2 中指定的节点模型，创建如图 L8.2 所示的网络拓扑。可在 **Object Palette** 的 *utilities* 文件夹中找到 *Packet Discarder* 节点模型。

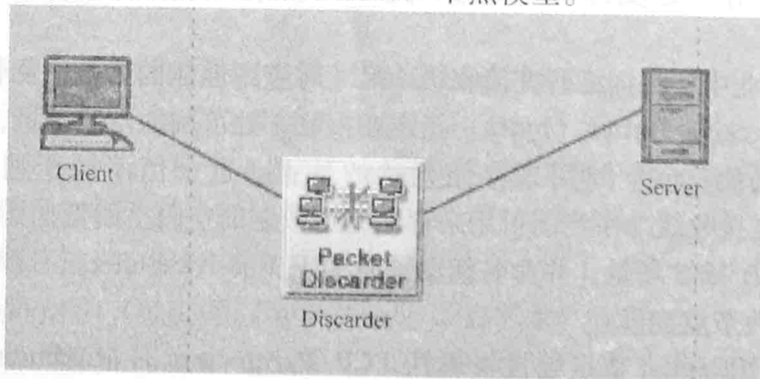


图 L8.2　TCP 拥塞控制场景的网络拓扑

表 L8.2 TCP 拥塞控制场景的网络拓扑摘要

对象名	对象模型
Client	*ppp_ wkstn* 节点对象
Server	*ppp_ server* 节点对象
Discarder	*Packet Discarder* 节点对象
网络中的所有链路	*PPP_ DS1* 链路对象

3）验证链路连通性。

4）添加 *Application Config* 和 *Profile Config* 节点对象。

5）创建一个 FTP 应用，称为 FTP，它每隔 5min 发出一条 get 命令，下载一个 2Mb 大小的文件。使用一个常数分布指定文件尺寸和请求间隔时间（见 5.4.3 节）。

6）配置一个概要，称为 FTP User，它如下运行 FTP 应用：

① 设置应用启动时间偏移为 1s（使用一个常数分布）。

② 设置概要启动时间为 100s（使用一个常数分布）。

③ 设置应用和概要可重复性为 Once at Start Time。

7）在网络中部署所定义的概要，使 Client 作为 FTP 应用的源，节点 Server 作为一台服务器。

8）如下指定节点配置：

① 在 Client 和 Server 节点处提升属性 **TCP... TCP Parameters**。

② 配置 Discarder 节点，如下丢弃报文：

- 在时刻 104s 和 104.5s 之间丢弃 1 条报文。
- 在时刻 106s 和 106.5s 之间丢弃 1 条报文
- 在时刻 108s 和 108.5s 之间丢弃 3 条报文

9）配置仿真收集如下统计量：

① **FTP** 节点统计类的 **Download Response Time**（**sec**）。

② **TCP Connection** 节点类的 **Congestion Window Size**（**bytes**）。确保这个统计量是使用 all values 模式收集的（见 4.2.7 节）。

③ **TCP Connection** 节点类的 **Retransmission Count**。

④ 所有 **point-to-point** 链路统计量。

10）设置提升的 **TCP... TCP Parameters** 属性为如下值：Tahoe、Reno、New Reno 和 SACK。

11）运行仿真 130s。

问题

Q15. 给出本项研究中所用每种 TCP 风格的简短描述。

Q16. 详细研究场景配置，并针对每种 TCP 风格，以表的形式指定 OPNET 配置属性 **Fast Retransmit**、**Fast Recovery** 和 **Selective ACK**（**SACK**）的值。该表应该包含具有

如下表头的三列：TCP 风格、属性名和属性值。简短地描述快速重传、选择性确认以及快速恢复的 Reno 和 New Reno 变种是如何工作的。

Q17. 详细研究为 Server 节点收集的统计量，画出该节点处 TCP 拥塞窗口尺寸的一个图形，并将仿真中从 102s 到 116s 的时间段放大。识别 TCP 过程的慢启动和拥塞避免阶段（即为每个阶段指定开始时间和结束时间）。

Q18. 复制在 Server 节点处 TCP 拥塞窗口尺寸的图形。放大从 103. 8s 到 104. 4s 的时间段。解释 TCP 的每种风格如何对单条报文丢失做出反应。

Q19. 复制在 Server 节点处 TCP 拥塞窗口尺寸的图形。放大从 105. 8s 到 106. 4s 的时间段。解释 TCP 的每种风格如何对第二条报文丢失做出反应。

Q20. 复制在 Server 节点处 TCP 拥塞窗口尺寸的图形。放大从 107. 8s 到 108. 4s 的时间段。解释 TCP 的每种风格如何对多条报文丢失做出反应。

Q21. 详细研究在 Server 节点处所收集的 TCP 重传计数统计量，并画出该节点处 TCP 重传计数的图形。您认为所收集的统计结果与所配置的报文丢弃行为一致吗？为什么一致或为什么不一致？

Q22. 详细研究 FTP 下载响应时间的收集统计量，并画出其图形。哪种 TCP 风格造成最低的 FTP 下载响应时间？为什么？哪种 TCP 风格造成最大的 FTP 下载响应时间？为什么？

Q23. 详细研究针对 Client 和 Discarder 节点之间链路上吞吐量所收集的统计量，并画出其图形。您认为所收集的仿真结果与所配置的报文丢弃和所观察到的拥塞窗口行为一致吗？为什么一致或为什么不一致？

实验室作业 9：IP 编址和网络地址转换

针对这项实验室作业，建议阅读第 1 ~7 章、第 9 章和第 10 章。

L9.1 引言

在这项实验室作业中，将采用子网分割，实现 IP 地址分配以及网络地址转换（NAT）设备的配置。具体而言，将开发并配置带有几个子网的一个网络的模型以及网络的接入网关中的一个 NAT 设备。

考虑如下情形。一家互联网服务提供商（ISP）将一个 IP 地址块销售给称为软件 R' Us（SRU）的一家本地公司。ISP 将如下 IP 地址范围分配给 SRU：197.13.45.128/27。依据如下需求（即需要的 IP 地址数量），SRU 网络管理员将所分配的 IP 地址分布在公司各部门间：

1）应用开发（AD）：12 个可用 IP 地址。

2）应用测试（AT）：4 个可用 IP 地址。

3）管理和财务（AA）：3 个可用 IP 地址。

在这项实验室作业中，将实施如下任务：

1）为 SRU 内的每个部门确定合适的 IP 地址分配。

2）将 SRU 网络分成子网。

3）以合适的 IP 地址“手工”（即不使用 Auto - Assign 选项）配置每个子网。

4）为 SRU 各子网之一在网关路由器上指定网络和端口地址转换规则。

将如下测试最终配置，方法是运行一个简单的仿真，并验证每个节点能够发送和接收数据。

L9.2 初步计算

在这部分实验室作业中，将为 SRU 的每个部门计算 IP 地址分配。假定每个部门被放置在其自己的子网内。注意，所要求的 IP 地址数量可能并不对应于所分配的 IP 地址数量。仅有可用地址可被分配给各节点。不可用的 IP 地址是这样的地址，它们具有特殊含义，如广播地址（即 IP 地址的主机部分为全 1）或这个（子）- 网络的地址（即 IP 地址的主机部分为全 0）。为完成这部分作业，需要回答如下问题：

问题

Q1. 卖给 SRU 多少个 IP 地址？这些地址中有多少是可用的（即可被指派给一个节

点)?

Q2. 对于每个 SRU 部门子网:

① 为满足部门的地址需求，应该分配多少个 IP 地址?

② 这个部门子网的网络地址是多少?

③ 这个部门子网的广播地址是多少?

④ 这个部门子网的掩码值是多少?

⑤ 在这个部门子网中第一个和最后一个可用 IP 地址是多少? 请使用无类域间路由(CIDR) 表示法。

⑥ 在这个部门子网中剩下的未用地址数是多少?

L9.3　仿真建立

在这部分作业中，将创建 SRU 网络的一个仿真模型。

1) 创建一个新项目和一个空场景，分别命名为 Assignment 09 和 IP_ Addressing (见 1.6.2 节)。

2) 使用表 L9.1 中指定的节点和链路模型，创建如图 L9.1 所示的网络拓扑 (见 2.3 节和 2.4 节)。

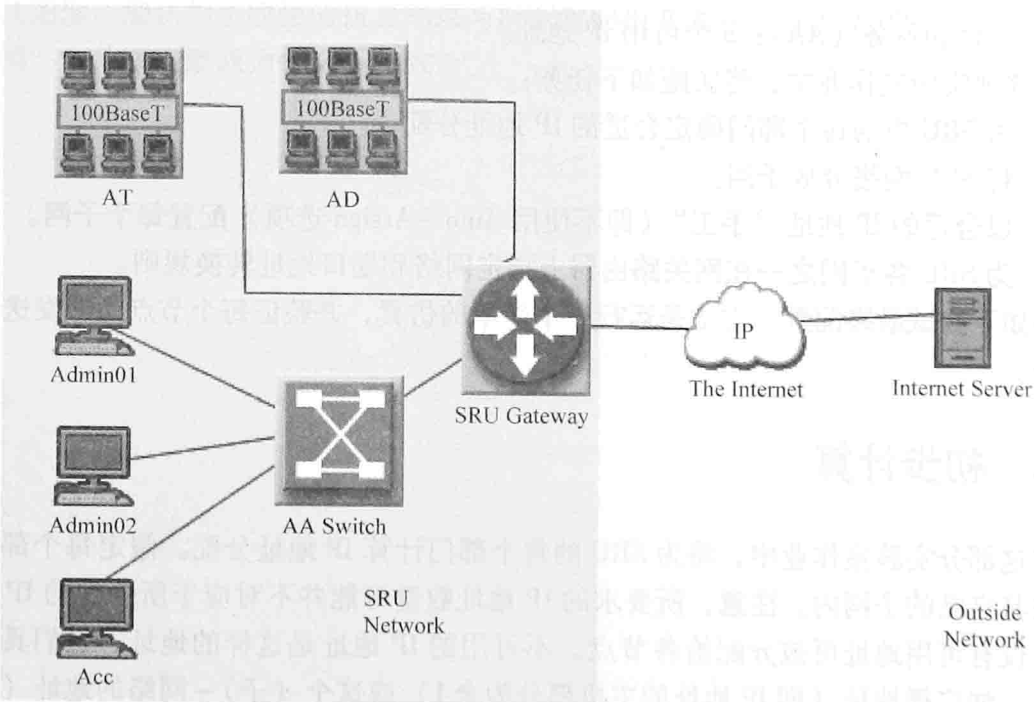


图 L9.1　这项作业的网络拓扑

表 L9.1 网络拓扑摘要

对象名	对象模型
AT、AD	100*BaseT_ LAN* 节点对象
SRU Gateway	*ethernet4_ slip8_ gtwy* 节点对象
Admin01、Admin02、Acc	*ethernet_ wkstn* 节点对象
AA Switch	*ethernet16_ switch* 节点对象
SRU 网络中的所有链路	100*BaseT* 链路对象
The Internet	*ip32_ cloud* 节点对象
Internet Server	*ppp_ server* 节点对象
外部网络中的所有链路 SRU Gateway < - > The Internet	*PPP_ DS3* 链路对象
所有 IP 流量需求	*ip_ traffic_ flow* 需求对象

3）验证链路连通性（见2.6.2节）。

4）配置 The Internet 云丢弃所有到达报文的0.01%，并为通过它的所有流量引入100ms 的额外时延。

5）设置IP地址和相应的子网掩码（见9.2节和9.3节）：

①设置 Internet Server 的IP地址和掩码为137.34.78.23/25。

②设置 The Internet 节点的接口上的IP地址和掩码：

- 到 Internet Server 的接口：137.34.78.24/25。
- 到 SRU Gateway 的接口：146.56.12.1/29。

③设置 SRU Gateway 节点的接口上的IP地址和掩码：

- 到 The Internet 的接口：146.56.12.2/29。
- 到 AD LAN 的接口：为AD子网分配的最后一个可用IP地址和相应的AD子网掩码值。
- 到 AT LAN 的接口：为AT子网分配的最后一个可用IP地址和相应的AT子网掩码值。
- 到 AA Switch 的接口：为AA子网分配的最后一个可用IP地址和相应的AA子网掩码值。

④设置 AD LAN 对象的IP地址和掩码为AD子网中的第一个可用地址和相应掩码值。

⑤设置 AT LAN 对象的IP地址和掩码为AT子网中的第一个可用地址和相应掩码值。

⑥设置 Admin01、Admin02 和 Acc 端节点的IP地址和掩码分别为AA子网分配的第一、第二和第三个可用地址及其相应的AA子网掩码。

6）配置 AD 和 AT LAN 对象分别有12和4台工作站（即属性 **LAN...Number of Workstations**）。

7）从 AD 和 AT LAN 到 Internet Server 部署两个 IP 流量需求对象。配置这些需求对象以速率 1.53Mbit/s（即一条 T1 线路的传输速率）和 200 报文数/s 产生流量，时长为 1h（见 6.6.1 节）。

8）从 Admin01、Admin02 和 Acc 节点到 Internet Server 部署三个 IP 流量需求对象。配置所有这些需求对象以速率 0.1Mbit/s 和 200 个报文数/秒产生流量，时长为 1h。

9）配置所有需求仅产生 1.0% 的显式流量（见 6.6.1 节）。

10）配置您的仿真收集所有 **Demand** 和所有链路 *point - to - point* 统计量（见 6.8 节和 4.2 节）。

11）设置全局仿真属性 **IP... IP Interface Addressing Mode** 为 Auto Addressed/Export（见 4.3 节）。

12）执行仿真 1h。

问题

Q3. 仿真运行产生了多少条 DES 日志消息？如果创建的 DES 日志消息总数超过 4，那么可能的情况是，您犯了一次配置错误。在那种情形中，仔细研究每条 DES 日志消息，识别错误，改正错误，并之后重新运行仿真。

Q4. 以在网络中所有活跃接口上的 IP 地址指派，详细研究由 OPNET 输出的文件（即 Assignment 09 - IP_ Addressing - DES - 1 - ip_ addresses. gdf）。请将这个文件与您的报告关联（见 9.3.7 节）。从所创建的 OPNET 仿真输出的 IP 地址指派必须匹配您的配置。

Q5. 创建一个图形，其中展示每个需求产生流量总量（以比特每秒为单位）。

Q6. 创建一个图形，其中展示网络中从每个端节点发送的流量总量（以比特每秒为单位）。存在不产生任何流量的节点吗？为什么有或为什么没有？（提示：确保详细检查了仿真中的所有端节点）。

Q7. 详细检查到达 Internet Server 节点的流量总量（以比特每秒为单位）。相比于网络中由所有流量需求产生的总流量，这个量如何？为什么？

L9.4 配置动态 NAT

在这部分作业中，将在 SRU Gateway 上配置一个 NAT 设备。假定 AD 部门增加了 10 倍，现在不能使每台个体机器被指派一个独立的全局 IP 地址。结果是，网络管理员决定在 SRU Gateway 中为 AD 子网配置 NAT。因为流量仅从 AD LAN 流向 Internet Server，但没有反向流量，所以仅为从 AD LAN 发出的报文指定 NAT。具体而言，将实施如下步骤，在 SRU Gateway 上配置 NAT：

1）复制当前场景，命名新场景为 IP_ Addressing_ wNAT。

2）将 AD LAN 上的 IP 地址改变为私网地址 192.168.0.1，并将子网掩码设置为 255.255.255.0。

3）在 SRU Gateway 中 SRU Gateway 和 AD LAN 之间接口上设置 IP 地址为 192.168.0.2，并设置网络掩码为 255.255.255.0。

4）改变 AD LAN 中的工作站数量为 120。

5）在 SRU Gateway 上配置 NAT 参数：

① 指定全局 IP 地址池，可用于 AD 子网中各节点的地址转换。称之为 AD pool（见 10.1.3 节）。

- 将接口名设置为 SRU Gateway 和 AD LAN 之间接口的名字。
- 将池中 IP 地址范围设置为最初分配给 AD 子网的地址范围（在 L9.1 节和 L9.2 节）。
- 设置地址池掩码为 AD 子网的掩码。
- 设置端口号为自动选择的。

② 定义转换规则，称之为 Local AD to Global AD（见 10.1.4 节）：

- 设置原报文的源 IP 地址为 192.168.0.1。
- 设置原报文的子网掩码为 255.255.255.0。
- 设置被转换报文的源 IP 地址为从以前定义的 AD 池动态选择的。

③ 指定和配置将在其上部署 NAT 规则的接口（见 10.1.5 节）：

- 将接口名设置为 AD LAN 和 SRU Gateway 之间接口的名字。
- 设置这个接口的状态为 inside。
- 配置接口使用 Local AD to Global AD 转换规则。

6）执行仿真 1h。

问题

Q8. 相比于网络中由所有 IP 需求产生的总流量，以比特每秒为单位表示的，到达 Internet Server 节点的流量总量情况如何？为什么？

Q9. 相比于在前一个场景中收集的结果，源自 AD LAN 的一个 IP 需求产生的流量总量（以比特每秒为单位）发生变化吗（即自 AD LAN 中的节点数量增加以来）？为什么发生变化或为什么不发生变化？

L9.5 配置端口地址转换

在这部分作业中，将在 SRU Gateway 上配置端口地址转换（PAT）。考虑这样一种情况，其中 AD 部门决定使其雇员运行 FTP 应用。部署这项应用，并观察发生的情况：

1）复制当前场景，命名新场景为 IP_ Addressing_ wPAT。

2）添加一个 *Application Config* 对象，并定义一个 High Load FTP 应用（见 5.3.1 节和 5.4.3 节）。

3）添加一个 *Profile Config* 对象，并创建一个新的用户概要，该概要运行所定义的 FTP 应用（见 7.2 节）。

4）部署所定义的概要，使 AD LAN 是 FTP 应用的源，Internet Server 是目的地（见 7.4 节）。

5）执行仿真。

问题

Q10. 仿真正常地终止吗？DES 日志产生任何警告或错误消息吗？如果是这样的话，DES 日志报告了什么种类的消息？为什么产生这些消息？您认为什么是问题的根本原因？

因为 FTP 运行在 TCP 之上，所以它要求连接的两端交换消息。我们的源 NAT 配置对 FTP 流量不能正常工作，原因是 SRU Gateway 没有被配置处理来自 Internet Server 的响应。因此，将需要指定另一个规则集，用于转换来自 Internet Server 的到达流量：

1）在 SRU Gateway 上配置 PAT 参数。

① 更新池配置，将转换私有 IP 地址改变为单个全局 IP 地址但带有不同的端口号：

- 将池中的 IP 地址改变为 AD 子网中的第一个地址。
- 将地址池的掩码设置为 AD 子网的掩码。
- 将端口号设置为 2000～3000 的范围。

② 添加另一条转换规则，称之为 Global to Local AD（见 10.1.4 节）：

- 将原报文的源 IP 地址设置为 Any 或 Internet Server 的 IP 地址。在后一种情形中，也希望将原报文的掩码设置为 Internet Server 的掩码。
- 设置 *original*（原）报文的目的地 IP 地址和掩码为 AD 子网的第一个地址和 AD 子网相应的掩码值。
- 设置被转换报文的目的地 IP 地址为 192.168.0.1（这是 AD LAN 对象的 IP 地址）。

③ 配置新转换规则将部署于其上的接口（见 10.1.5 节）：

- 添加另一个 **Interface Information** 行。
- 设置接口的名字为 SRU Gateway 和 The Internet 之间接口的名字。
- 设置这个接口的状态为 outside。
- 配置接口使用 Global to Local AD 转换规则。

2）执行仿真 1h。

问题

Q11. 仿真正常终止吗？注释：可忽略出现在仿真结束时的一些 DES 日志消息，这些消息声称“The IP routing table on this node does not have a route to the destination X. X. X. X”（在这个节点上的 IP 路由表没有到目的地 X. X. X. X 的一条路由）或任何其他相关的消息。这看起来是某个内部 OPNET 缺陷，对仿真具有非常小的影响。

Q12. 在 AD LAN 和 SRU Gateway 之间的链路上的吞吐量是多少（以比特每秒为单

位表示)?

Q13. 存在任何到达AD LAN的流量吗？为什么有或为什么没有？

Q14. 相比于由SRU Gateway转发到the Internet的总流量（以比特每秒为单位表示），从AD、AT和AA子网到达SRU Gateway的总流量（以比特每秒为单位表示）情况如何？

实验室作业10：提供服务质量支持

针对这项实验室作业，建议阅读第1~7章、第9章和第10章。

L10.1 引言

在这项实验室作业中，将深入探索在网络中部署各种服务质量（QoS）机制的技术。特别地，将深入探索通过诸如随机早期检测（RED）、加权的RED（WRED）、加权的公平排队（WFQ）和具有低延迟队列的WFQ（WFQ-LLQ）等机制，如何提供各种等级的服务。

L10.2 设置基线场景

在这部分作业中，将建立一个基线场景，其中四种不同的流量类在没有QoS支持的网络中竞争有限的资源。

1）创建一个新项目和一个空场景，分别命名为Assignment 10和FIFO（见1.6.2节）。

2）使用表L10.1中指定的节点和链路模型，创建如图L10.1所示的网络拓扑（见2.3节和2.4节）。

表L10.1 网络拓扑摘要

对象名	对象模型
Client（ToS 0） Client（ToS 1） Client（ToS 2） Client（ToS 3）	*ppp_ wkstn* 节点对象
Destination	*ppp_ server* 节点对象
Router	*ethernet4_ slip8_ gtway* 节点对象
网络中的所有链路	*PPP_ DS3_ int* 链路对象
网络中的所有需求对象	*ip_ traffic_ flow* 需求对象

3）验证链路连通性（见2.6.2节）。

4）添加一个 *QoS Attribute Config* 对象。

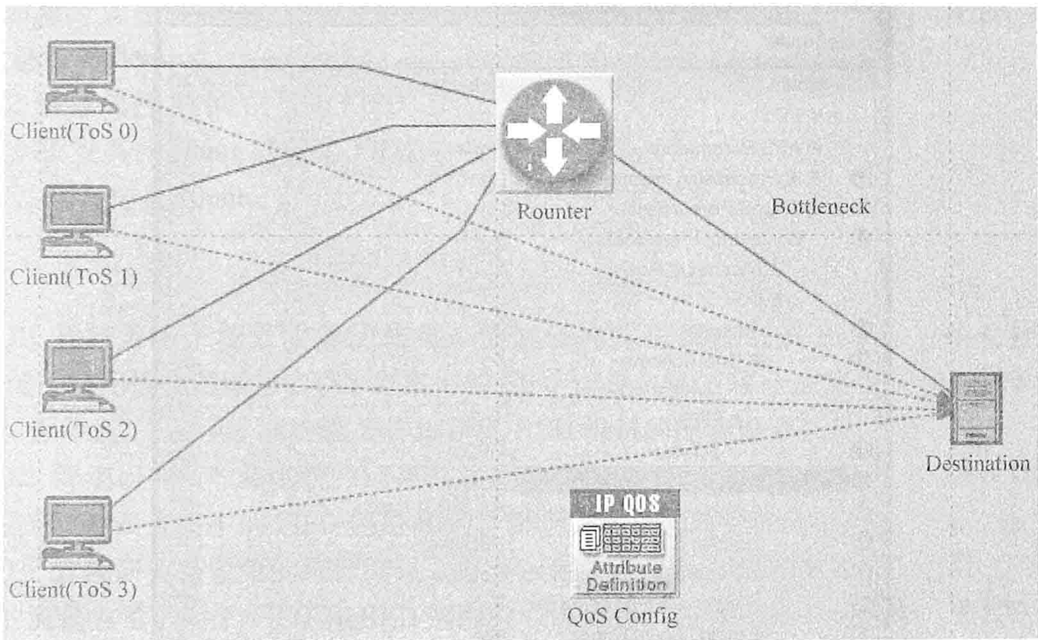


图 L10.1 网络拓扑

5）如下添加和配置所有需求对象：

① 以 1.53Mbit/s（即 T1 线路）传输数据，属性 **Traffic（bits/sec）**。

② 以 200 报文数/s 传输数据，属性 **Traffic（packets/sec）**。

③ 显式地对所有流量建模，设置属性 **Traffic Mix** 为 All Explicit。

6）通过设置属性 **Traffic Characteristics... Type of Service**，指定每个需求对象携带的流量类型（见 6.6 节）：

① Client（ToS 0）－－＞Destination 需求携带 Best Effort（0）流量。

② Client（ToS 1）－－＞Destination 需求携带 Background（1）流量。

③ Client（ToS 2）－－＞Destination 需求携带 Standard（2）流量。

④ Client（ToS 3）－－＞Destination 需求携带 Excellent Effort（3）流量。

7）设置 Router 和 Destination 之间链路上的数据速率为 5Mbit/s。

8）配置 Router 运行 FIFO 排队概要（见 10.4.3 节）：

① 确定附接到 Router 和 Destination 之间链路的 Router 接口数量。

② 配置标志的接口使用默认 FIFO QoS 概要。保持其他配置属性设置为其默认值。图 L10.2 给出 Router 节点的一种可能配置例子，它支持 FIFO QoS 方案。

9）配置仿真收集如下统计量：

① 需求统计类中的所有统计量。

② 类 **IP interface** 的如下节点统计量：

- **Buffer Usage（bytes）**
- **Queue Delay Variation（sec）**

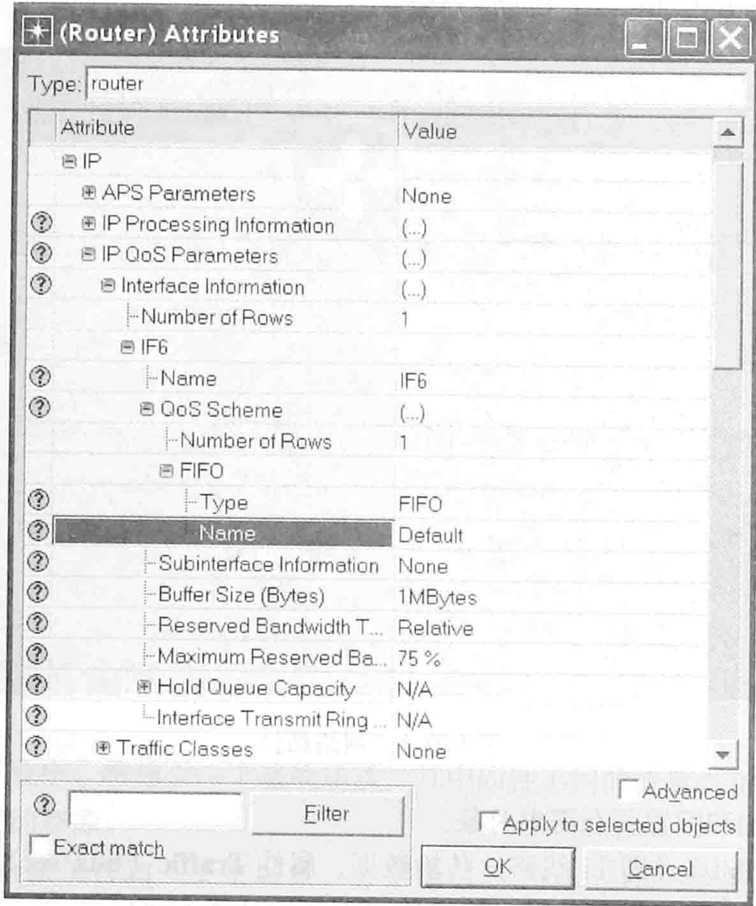


图 L10.2　支持 FIFO QoS 方案的 Router 配置例子

- **Queuing Delay**（**sec**）
- **Traffic Dropped**（**bits/sec**）
- **Traffic Sent**（**bits/sec**）
- **Traffic Received**（**bits/sec**）

10）运行仿真10min。

L10.3 FIFO 和 RED 比较

在这部分作业中，将比较 FIFO 和 RED QoS 方案如何影响通过网络携带的流量。

1）复制当前场景，命名新场景为 RED（见 1.7 节）。

2）如下创建称为 FIFO w. RED 的一个新 FIFO QoS 概要：

① 在 QoS Attribute Config 节点中添加称为 FIFO w. RED 的一个新 FIFO 概要。

② 配置 FIFO w. RED 概要支持默认 RED 配置（即设置属性 **RED Parameters** 为 RED）。

3）在附接到 Router 和 Destination 之间链路的 Router 接口上，如下部署 FIFO w.

RED 概要：

① 展开属性 **IP... IP QoS Parameters... Interface Information... <interface name>... QoS Scheme...**

② 设置属性 **Type** 的值为 FIFO。

③ 设置属性 **Name** 的值为 FIFO w. RED。

4）运行仿真 10min。

问题

Q1. 比较为每个场景收集的结果。详细研究并显示在节点 Router 上外发接口上 FIFO 缓冲使用情况的图形。QoS 方案中的 FIFO 还是 RED，哪个方案得到较低的队列占有率？为什么？（提示：可能需要详细研究 RED 机制的配置设置）。

Q2. 详细研究所收集的统计结果，并显示在节点 Router 的外发接口上 FIFO 排队时延的图形。QoS 方案中的 FIFO 还是 RED，哪个方案得到较低的排队时延？为什么？

Q3. 详细研究所收集的统计结果，并显示在节点 Router 的外发接口上接收、发送和丢弃的流量总量。对于 FIFO 和 RED 这两种 QoS 方案，这些结果有何区别？为什么？

Q4. 详细研究针对任何需求流的报文所经历端到端时延所收集的统计结果。QoS 方案中的 FIFO 还是 RED，哪个方案得到较低的端到端时延？为什么？

L10.4 加权的 RED

在这部分作业中，将详细研究 WRED QoS 方案如何影响网络中的流量断续流。

1）复制 RED 场景，并命名新场景为 WRED。

2）如下创建称为 FIFO w. WRED 的一个新 FIFO QoS 概要：

① 在 QoS Attribute Config 节点中添加称为 FIFO w. WRED 的一个新 FIFO 概要。

② 配置 FIFO w. WRED 概要支持默认的 WRED 配置（即设置属性 **RED Parameters** 为 WRED）。

③ 通过设置属性 **Mark Probability Denominator** 的值为 1、5、10、20，配置 WRED 以一个不同的概率丢弃每个流类的过量报文。回顾一下，可将这个属性直接设置为多个值（以逗号分隔这些值）。

3）在附接到 Router 和 Destination 之间链路的 Router 接口上，部署 FIFO w. WRED 概要。

4）运行仿真 10min。

问题

Q5. 详细研究所收集的统计结果，并针对每个流量需求断续流由目的地节点接收的流量总量显示一幅图形。ToS 0、ToS 1、ToS 2 或 ToS 3 中哪个流量类交付最大的流量？为什么？

Q6. 详细研究所收集的统计结果，并针对节点 Router 的外发接口上 FIFO 缓冲使用情况显示一幅图形。FIFO、RED 或 WRED 中哪个 QoS 方案得到最低的队列占有率？为什么？

Q7. 详细研究所收集的统计结果，并针对节点 Router 的外发接口上 FIFO 排队时延显示一幅图形。FIFO、RED 或 WRED 中哪个 QoS 方案得到最低的排队时延？为什么？

Q8. 详细研究所收集的统计结果，并针对节点 Router 的外发接口上接收、发送和丢弃的流量显示图形。FIFO、RED 或 WRED 三种 QoS 方案，这些结果有何区别？为什么？

Q9. 详细研究针对任何需求断续流的报文所经历的端到端时延所收集的统计结果。FIFO、RED 或 WRED 中哪个 QoS 方案得到最低的端到端时延？为什么？

Q10. 详细研究所收集的统计结果，并为每个类的流量需求所经历的报文抖动显示一幅图形。ToS 0、ToS 1、ToS 2 或 ToS 3 中哪个流量类经历最低的报文抖动？为什么？FIFO、RED 或 WRED 中哪个 QoS 方案得到流量类 ToS 3 的最低报文抖动？为什么？

L10.5 加权的公平排队

在这部分作业中，将详细研究 WFQ QoS 方案如何影响网络中的流量断续流。

1）复制当前场景，并命名新场景为 WFQ。

2）打开 *QoS Attribute Config* 对象，并如下创建称为 4 ToS Classes 的一个新的 WFQ QoS 概要：

① 复制称为 ToS Based 的 WFQ 概要，方法是右击该概要的标题，并从出现的菜单中选择 **Duplicate Row** 选项。

② 将新概要的名字设置为 4 ToS Classes。

③ 通过设置属性 **Number of Rows** 的值为 4，去除新概要中默认 8 个类中的 4 个。

④ 详细检查每个类的配置，并验证剩下的 4 个类仅接受分别标记为 ToS 值 0、1、2 和 3 的报文。

3）在附接到 Router 和 Destination 的链路上的 Router 接口上，部署 4 个 ToS Classes。

① 展开属性 **IP... IP QoS Parameters... Interface Information... <interface name>... QoS Scheme...**。

② 设置属性 **Type** 的值为 WFQ（Classes Based）。

③ 设置属性 Name 的值为 4 ToS Classes。

4）运行仿真 10min。

问题

Q11. 详细研究所收集的统计结果，并显示一幅图形，其中展示在节点 Router 的外发接口上每个 WFQ 子队列中的缓冲使用情况。ToS 0、ToS 1、ToS 2 或 ToS 3 中哪个流量类具有最低缓冲使用率？哪个流量类具有最高缓冲使用率？为什么？

Q12. 详细研究所收集的统计结果，并显示一幅图形，其中展示在节点 Router 的外

发接口上每个子队列中的排队时延？ToS 0、ToS 1、ToS 2 或 ToS 3 中哪个流量类具有最低排队时延？哪个流量类具有最高排队时延？为什么？

Q13. 详细研究所收集的统计结果，并显示一幅图形，其中展示在节点 Router 的外发接口上每个子队列接收、发送和丢弃的流量总量。ToS 0、ToS 1、ToS 2 或 ToS 3 中哪个流量类具有最低的接收/发送/丢弃的流量？哪个流量类具有最高的接收/发送/丢弃的流量？为什么？

L10.6 具有低延迟队列的 WFQ

在这部分作业中，将详细研究将 WFQ 子队列之一配置为一个低延迟队列如何影响 WFQ QoS 方案中的资源分布。

1）复制 WFQ 场景，命名新场景为 WFQ_ LLQ。

2）如下修改 WFQ QoS 概要 4 ToS Classes：指定存储背景流量（即流量类 ToS 1）的队列为一个低延迟队列，方法是将其属性 **Queue Category** 的值设置为 Low Latency Queue。

3）运行仿真 10min。

问题

Q14. 针对新场景回答问题 Q11 ~ Q13，其中流量类 ToS 1 的 WFQ 队列被指定为一个低延迟队列。相比于针对 WFQ 场景收集的结果，您观察到了流量类间资源分布（如缓冲使用率、排队时延和接收/发送/丢弃的流量总量）发生了哪些变化？为什么？

L10.7 改变 WFQ 配置

在这部分作业中，将详细研究修改 WFQ QoS 方案的默认配置如何影响在 WFQ 中的资源分布。

1）复制 WFQ 场景，命名新场景为 WFQ_ Variations。

2）如下修改 WFQ QoS 概要 4 ToS Classes：修改 WFQ 子队列［缓冲极大尽力流量（Excellent Effort）（即流量类 ToS 3）］的权重为 80。

3）如下配置 Router 的外发接口为 QoS 流量仅保留 40% 的链路带宽：设置属性 **IP... IP QoS Parameters... Interface Information... < interface name >... Maximum Reserved Bandwidth** 的值为 40。

4）运行仿真 10min。

问题

Q15. 针对新场景，回答问题 Q11 ~ Q13，其中修改 WFQ QoS 方案的默认配置。相比于针对 WFQ 场景收集的结果，您观察到流量类间资源分布（如缓冲使用率、排队时延和接收/发送/丢弃的流量总量）发生了哪些变化？为什么？

实验室作业11：采用RIP进行路由

针对这项实验室作业，建议阅读第1~4章、第9章和第11章。

L11.1 引言

在这项实验室作业中，将详细研究在各种配置状况下路由信息协议（RIP）的性能。特别地，将详细研究在使用RIP路由的一个网络中路由变化有多快地进行传播。另外，将实践创建复杂的OPNET仿真模型，配置RIP，失效和恢复网络中的链路，修改统计收集模式，输出OPNET报告，等等。

L11.2 建立初始RIP场景

在这部分实验室作业中，将为详细研究RIP性能而创建一个网络的仿真模型：

1）创建一个新项目和一个空场景，分别命名为Assignment 11和RIP Normal（见1.6.2节）。

2）使用表L11.1中指定的节点和链路模型，创建如图L11.1所示的网络拓扑（见2.3节和2.4节）。

表L11.1 RIP研究的网络拓扑摘要

对象名	对象模型
Router 1 ~ Router 8	*ethernet4_ slip8_ gtwy* 节点对象
Host 1 ~ Host 4	*ppp_ wkstn* 节点对象
网络中的所有链路	*PPP_ DS1* 链路对象

3）验证链路连通性（见2.6.2节）。

4）在网络中所有活跃接口上部署IPv4地址（即下拉菜单选项**Protocols→IP→Addressing→Auto - Assign IPv4 Addresses**，见9.3.4节）。

5）配置RIP为网络中所有接口上的一个路由协议（即选项**Protocols→IP→Routing→Configure Routing Protocols**，见11.1.1节）。

6）在网络中所有路由器接口上设置RIP通告模式为No Filtering（见11.2.3节）：

①选择网络中的所有路由器。

②在任何被选中节点上展开属性**IP Routing Protocols... RIP Parameters... Interface Information**。

③在节点的所有接口上设置属性**Advertisement Mode**的值为No Filtering。

④选择检查框*Apply to selected objects*，并单击**OK**按钮，保存网络中所有被选中节

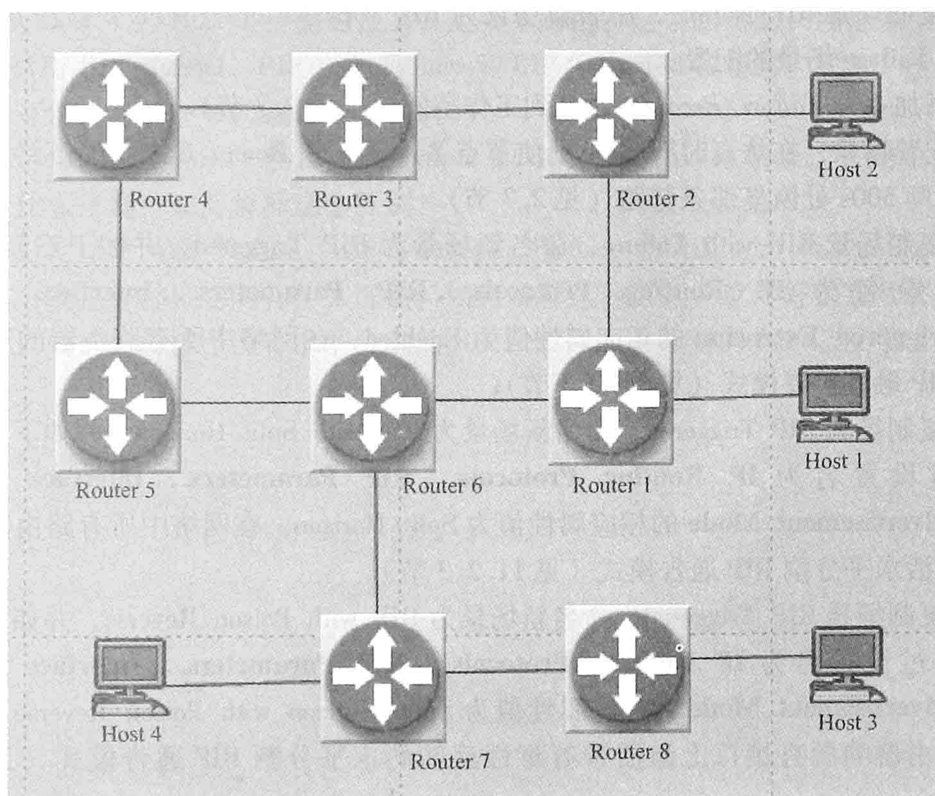


图 L11.1　RIP 研究的网络拓扑

点上的配置改变。

7）通过设置全局属性 **Simulation Efficiency... RIP Sim Efficiency** 为 Disabled，禁止 RIP 仿真效率模式（见 11.2.5 节）。

8）通过设置全局仿真属性 **IP... IP Interface Addressing Mode** 为 Auto Addressed/Export，配置仿真输出 IP 地址分配（见 9.3.7 节）。

9）通过设置属性 **Reports... RIP Routing Table** 的值，配置所有路由器在仿真时间 400s 处和在仿真结束处输出它们的路由表（见 9.6.3 节）。

10）配置仿真收集如下统计量：

① 类 **RIP** 中的全局统计量 **Traffic Received (bits/sec)**。将这个统计量的收集模式改变为 all values（见 4.2.7 节）。

② 类 **RIP** 的全局统计量 **Convergence Duration**。确保这个统计量的收集模式也设置为 all values，而画线风格设置为 discrete（见 4.2.6 节和 4.2.7 节）。

11）执行场景 15min，并验证其运行没有任何错误。

L11.3　配置其他 RIP 场景

在这部分作业中，将为评估不同配置下 RIP 的性能，创建几个附加场景。

1）复制场景 RIP Normal，命名新场景为 RIP with Failure（见 1.7 节）。如下更新 RIP with Failure 场景的配置：

① 添加一个 *Failure/Recovery* 对象到工作空间中（见 2.7 节）。

② 配置场景，在仿真时间 300s 处使节点 Router 1 和 Router 6 之间的链路失效，并在仿真时间 500s 处恢复那条链路（见 2.7 节）。

2）复制场景 RIP with Failure，命名新场景为 RIP Triggered，并如下更新其配置：通过设置名为 **IP Routing Protocols... RIP Parameters... Interface Information... Triggered Extension** 的相应属性值为 Enabled，在网络中所有路由器的所有接口上激活 RIP 触发扩展模式（见 11.2.3 节）。

3）复制场景 RIP Triggered，命名新场景为 RIP with Split Horizon，并如下更新其配置：通过设置名为 **IP Routing Protocols... RIP Parameters... Interface Information... Advertisement Mode** 的相应属性值为 Split Horizon，在网络中所有路由器的所有接口上激活水平分割 RIP 通告模式（见 11.2.3 节）。

4）复制场景 RIP Triggered，命名新场景为 RIP with Poison Reverse，并如下更新其配置：通过设置名为 **IP Routing Protocols... RIP Parameters... Interface Information... Advertisement Mode** 的相应属性值为 Split Horizon with Poison Reverse，在网络中所有路由器的所有接口上激活带有毒性反转的水平分割 RIP 通告模式（见 11.2.3 节）。

5）执行所有场景 15min，并详细研究所收集的结果。

问题

Q1. 通过检查带有输出 IP 地址（由仿真创建的）的文件，详细研究场景 RIP with Failure 的网络中的 IP 地址分配。

Q2. 详细研究仿真结果，并显示如下统计量的图形：

1）针对 RIP Normal、RIP with Failure 和 RIP Triggered 场景，在独立的平板中画出 **RIP... Traffic Received（bits/sec）**。

2）针对上面三种场景，在相同的平板中以堆叠统计模式画出 **RIP... Convergence Duration**。

3）针对 RIP with Split Horizon 和 RIP with Poison Reverse 场景，在相同的平板中以堆叠统计模式画出 **RIP... Convergence Duration**。

Q3. 详细研究 RIP Normal 场景中所有路由器上仿真结束时的路由表，并回答如下问题：

1）从 Host 2 到 Host 3 传输的报文，走哪条路由？这条路由的长度是多少？

2）从 Host 4 到 Host 1 传输的报文，走哪条路由？这条路由的长度是多少？

Q4. 详细研究 RIP with Failure 场景中所有路由器上时间 400s 处的路由表，并回答如下问题：

1）从 Host 2 到 Host 3 传输的报文，走哪条路由？这条路由的长度是多少？

2）从 Host 4 到 Host 1 传输的报文，走哪条路由？这条路由的长度是多少？

Q5. 针对 RIP Normal、RIP with Failure 和 RIP Triggered 各场景，在 RIP 所接收流量总量之间有什么区别？如何解释这些区别？

Q6. 针对 RIP Normal、RIP with Failure 和 RIP Triggered 各场景，在路由表收敛时长之间有什么区别？如何解释这些区别？

Q7. 针对 RIP with Failure、RIP with Split Horizon 和 RIP with Poison Reverse 各场景，在路由表收敛时长之间有什么区别？如何解释这些区别？就对好消息和坏消息响应的快速性方面，如何排序没有水平分割的 RIP、带有水平分割的 RIP 以及带有毒性反转的 RIP？

实验室作业 12：采用 OSPF 进行路由

针对这项实验室作业，建议阅读第 1～7 章、第 9 章和第 11 章。

L12.1 引言

在这项实验室作业中，将详细研究 OSPF 路由协议的行为。特别地，将开发带有 OSPF 路由的一个网络的模型，观察使用的路由，并研究 IP 的负载均衡功能特征。另外，将网络域分成层次区，研究这种分隔如何影响网络中的路由。

L12.2 建立初始 OSPF 场景

在这部分实验室作业中，将为研究 OSPF 性能，创建一个网络的仿真模型：

1）创建一个新项目和一个空场景，分别命名为 Assignment 12 和 OSPF Flat（见 1.6.2 节）。

2）使用表 L12.1 中指定的节点和链路模型，创建如图 L12.1 所示的网络拓扑（见 2.3 节和 2.4 节）。

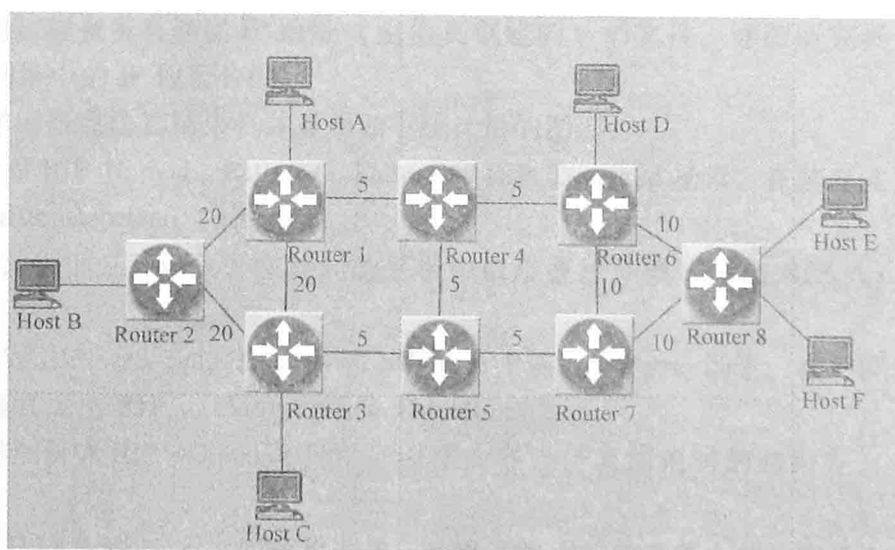


图 L12.1 OSPF 研究的网络拓扑

3）在网络中所有活跃接口上部署 IPv4 地址（即下拉菜单选项 **Protocols→IP→Addressing→Auto - Assign IPv4 Addresses**，见 9.3.4 节）。

4）配置 OSPF 为网络中所有接口上的一个路由协议（即选项 **Protocols→IP→Routing→Configure Routing Protocols**，见 11.1.1 节）。

表 L12.1　OSPF 研究的网络拓扑摘要

对象名	对象模型
Router 1 ~ Router 8	*ethernet4_ slip8_ gtwy* 节点对象
Host A ~ Host F	*ethernet_ wkstn* 节点对象
路由器之间的链路	*PPP_ DS3* 链路对象
主机和路由器之间的链路	100*BaseT* 链路对象

5）在表 L12.2 中给出的路由器之间的链路上配置成本（即使用选项 **Protocols→OSPF→Configure Interface Cost...**，见 11.3.6 节）。

表 L12.2　链路成本值摘要

链路名	链路成本	OSPF 区
Router 1 ~ Router 3 Router 1 ~ Router 2 Router 2 ~ Router 3	20	1
Router 1 ~ Router 4 Router 4 ~ Router 5 Router 4 ~ Router 3 Router 4 ~ Router 6 Router 5 ~ Router 7	5	0
Router 6 ~ Router 7 Router 6 ~ Router 8 Router 7 ~ Router 8	10	2

6）在如下主机节点对之间部署 IP 流量需求（即需求模型 *ip_ traffic_ flow*），并依据表 L12.3 中的规格配置它们：

- Host A 和 Host C
- Host C 和 Host D
- Host B 和 Host E
- Host B 和 Host F

表 L12.3　IP 需求配置参数摘要

属性	值
Traffic (bits/second)	T1_ 1hour_ bps
Traffic (packets/second)	200_ pps
所有剩余属性	默认值

7）通过设置全局仿真属性 **IP... IP Interface Addressing Mode** 为 Auto Addressed/Export，配置仿真输出 IP 地址分配（见 9.3.7 节）。

8）通过设置属性 **Reports... OSPF Routing Table** 和 **Reports... OSPF Link State Database** 的值为 Export at End of Simulation（见 9.6.3 节），配置所有路由器在仿真结束时输出其路由表和链路状态数据库。

9）通过设置全局属性 **Simulation Efficiency... OSPF Sim Efficiency** 的值为 Disabled，禁止 OSPF 仿真效率模式（见 11.3.11 节）。

10）执行场景 10min。

问题

Q1. 通过检查带有输出 IP 地址（由仿真创建的）的文件，详细研究网络中的 IP 地址分配。

Q2. 仔细检查任何一台路由器节点上链路状态数据库，并确保所有链路成本是要配置的那样。如果它们没有像配置的那样，那么应该纠正它们，并重新运行场景。在链路状态数据库，用于主机和路由器之间链路所用的成本是多少？

Q3. 在链路状态数据库中，识别 Router 3 和 Router 6 的表项。对于这些表项，为表中所示的每个链路 ID，识别邻接路由器的名字。

Q4. 详细研究各路由器的路由表项，为从 Host A 到 Host C 和从 Host C 到 Host D 的流量确定路由。为这些路由，列出每一跳的接口号和路由器名、每跳的链路成本和该路由的总成本。

Q5. 对以前配置的四个需求之报文穿越的路由可视化（即使用选项 **Protocols→IP→Demands→Display Route for Configured Demand** 或 *Ctrl + Alt + D* 键）。一次查看一个需求的路由，一般而言是有用的，即在 **Route Report for IP Traffic Flows** 窗口中倒换（toggle）属性字段 **Display** 的值。同样，确保在关闭之前，关闭路由显示。为需求 Host A 到 Host C 和 Host C 到 Host D 可视化的路由与在问题 Q4 中计算的是相同还是不同？

Q6. 列出为从 Host B 到 Host E、Host B 到 Host F、Host E 到 Host B 以及 Host F 到 Host B 的流量可视化各路由。

L12.3 带有基于报文的负载均衡选项的 OSPF

在这部分作业中，将详细研究基于报文的负载均衡如何影响流量路由：

1）复制场景 OSPF Flat，命名新场景为 OSPF with packet - based load balancing（见 1.7 节）。

2）改变 Router 2 的配置，使其采用基于报文的负载均衡选项（即改变属性 **IP... IP Routing Parameters... Load Balancing Options** 的值为 Packet - Based）。

3）重新运行仿真 10min，并详细研究所收集的结果。

问题

Q7. 可视化各种需求采用的路由。列出观察到的从 Host B 到 Host E、Host B 到 Host F、Host E 到 Host B 以及 Host F 到 Host B 的流量所经的路由。就这些路由而言，解释您认为最重要的是什么。这些路由与场景 OSPF Flat 中的相应路由有何区别？

Q8. 基于对负载均衡选项的理解，讨论对这些路由的观察是否与负载均衡的工作方式一致。

L12.4 采用层次结构选项的 OSPF

在这个场景中，将详细研究 OSPF 性能如何受到将网络分成多个区做法的影响：

1）复制场景 OSPF Flat，命名新场景为 OSPF Hierarchical。

2）将网络分成如表 L12.2 所示的各区（即使用选项 **Protocols→OSPF→Configure Areas**，见 11.3.8 节）：

① 1 区将由左侧的三条链路组成，成本为 20，每条链路与到 Host A、Host B 和 Host C 的链路连接。

② 2 区将由右侧的三条链路组成，成本为 10，每条链路与到 Host D、Host E 和 Host F 的链路连接。

③ 骨干区 0 将由中间的各条链路组成，都带有成本 5。

3）通过可视化网络的层次结构，验证区配置（即使用选项 **View→Visualize Protocol Configuration→OSPF Area Configuration...**）

4）重新运行仿真 10min，并详细研究所收集的结果。

问题

Q9. 观察用于所有经配置的需求的路由。在这个场景中的路由与用于 OSPF Flat 场景中的路由有何区别？您能解释区别的原因吗？

L12.5 采用区边界路由器的 OSPF

在这个场景中，将详细研究 OSPF 性能如何受到区边界路由器（ABR）存在的影响：

1）复制场景 OSPF Hierarchical，命名新场景为 OSPF with ABR。

2）配置 Router 3，隐藏 Host A、Host B 和 Host C 的子网地址，并防止这些地址通告到 0 区、1 区等之外（见 11.3.9 节）。

3）重新运行仿真 10min，并详细研究所收集的结果。

问题

Q10. 观察所有经配置的需求使用的路由。在这个场景中的路由与 OSPF Hierarchical

场景中使用的那些路由有何区别？您能解释这些区别的原因吗？

Q11. 哪台路由器是 1 区的区边界路由器？

Q12. 在 Router 3 处有穿过区 0 到区 1 的一些路由吗？在 Router 3 处有穿过区 1 到区 0 的一些路由吗？如果有，解释为什么会发生这种情况。

Q13. 存在一些不对称的路由吗？如果有，为什么？

最后：

1）复制场景 OSPF with ABR，命名新场景为 OSPF with ABR2。

2）配置 Router 6，隐藏 Host D、Host E 和 Host F 的子网地址，并防止它们通告到 2 区之外（见 11.3.9 节）。

3）重新运行仿真 10min，并详细研究所收集的结果。

问题

Q14. 观察所有经配置的需求所用的路由。在这个场景中的路由与 OSPF Hierachical 和 OSPF with ABR 场景中的那些路由有何区别？您能解释这些差异的原因吗？

Q15. 哪台路由器是 2 区的区边界路由器？

实验室作业 13：以太网

针对这项实验室作业，建议阅读第 1 ~4 章和第 12 章。

L13.1 引言

在这项实验室作业中，将探究总线和星形网络拓扑中以太网的性能。另外，这项实验将形象地说明如何使用 OPNET 的 **Rapid Configuration** 功能特征以及如何收集标量统计量和建立参数化的研究。

L13.2 总线拓扑

首先建立和配置一个总线网络，将用之评估以太网性能。

1）创建一个新项目和一个空场景，分别命名为 Assignment 13 和 Ethernet_ Bus（见 1.6.2 节）。当使用 Startup Wizard 创建一个新场景时，使用 Office 网络规模，并包括 ethcoax 模型族。

2）使用 OPNET 的 **Rapid Configuration** 功能特征［可通过 **Topology→Rapid Configuration...** 菜单选项访问（见 2.5 节）］，创建如图 L13.1 所示的网络拓扑。

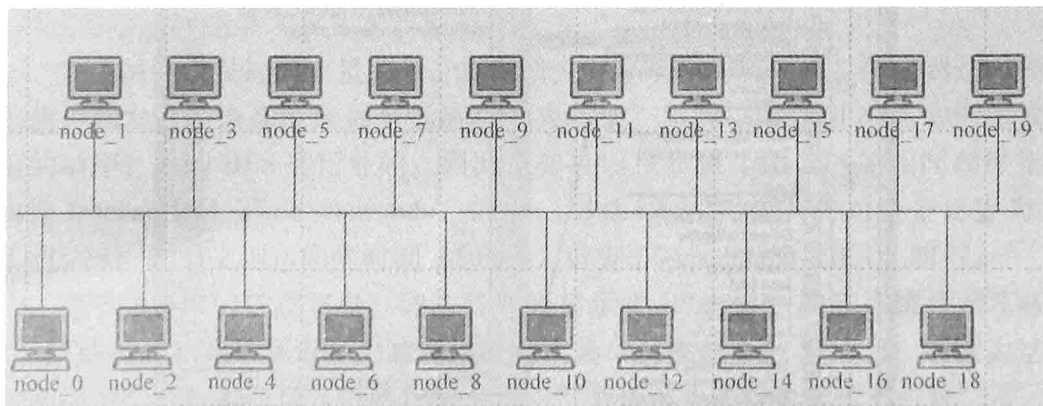


图 L13.1 以太网总线拓扑

① 如下配置总线拓扑中的节点和链路模型：

- **Node model**：ethcoax_ station
- **Number**：20
- **Link model**：eth_ coax
- **Tap model**：eth_ tap

② 保留其他属性设置为其默认值。

3）验证链路连通性（见 2.6.2 节）。

4）如下修改总线模型的属性：

① 改变模型名为 eth_ coax_ adv。

② 设置 **data rate** 为 500000bit/s。

③ 设置 **delay** 为 0.05s。回顾一下，**delay** 是一个高级属性，在可查看并修改其值之前，要求选择 *Advanced* 检查框。

5）配置网络中的所有节点，使之依据如下规格产生流量。回顾一下，可使用 **Select Similar Nodes** 选项和 *Apply to selected objects* 检查框，同时配置网络中的所有节点。如下配置 **Traffic Generation Parameters** 复合属性：

① 当设置属性值时，使用常数分布。

② 设置属性 **ON State Time（seconds）** 为 100s。

③ 设置属性 **OFF State Time（seconds）** 为 0s。

④ 提升属性 **Packet Generation Arguments... Interarrival Time（seconds）**。

⑤ 保留所有其他属性设置为其默认值。

6）配置仿真收集如下统计量：

① 类 **Traffic Sink** 中的全局统计量 **Traffic Received（bits/sec）**。通过选择 *Generate scalar data* 检查框并使用 *sample mean* 函数，配置这个统计量也作为使用样本均值的一个标量数据进行收集。欲了解期望的配置，见图 L13.2。

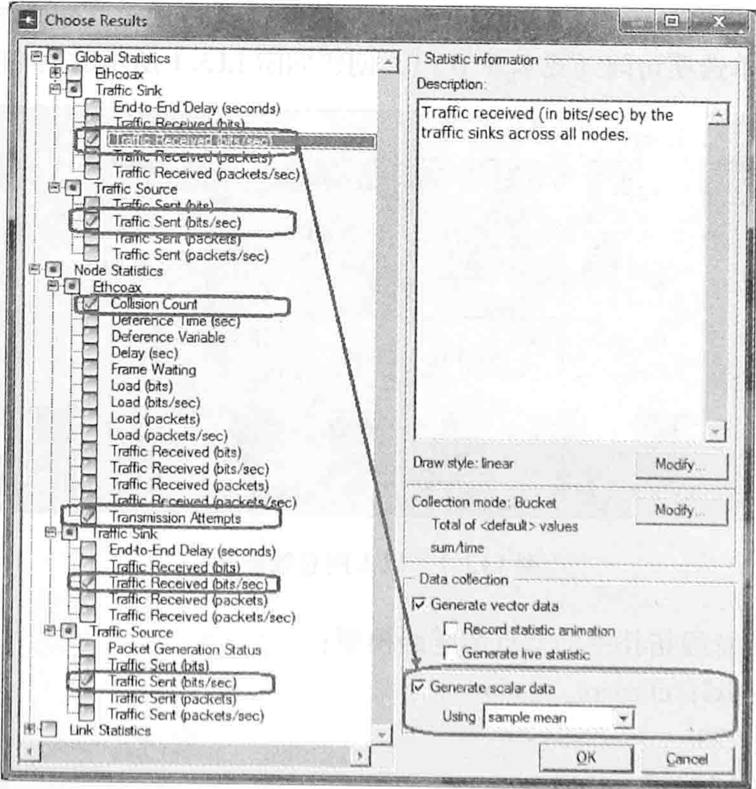


图 L13.2 配置仿真收集标量统计量

② 类 **Traffic Source** 中的全局统计量 **Traffic Sent**（bits/sec）：配置这个统计量也作为使用样本均值的一个标量数据进行收集。

③ 类 **Ethcoax** 中的如下节点统计量：**Collision Count**、**Transmission attempts**。

④ 类 **Traffic Source** 中的节点统计量 **Traffic Sent**（**bits/sec**）。

⑤ 类 **Traffic Sink** 中的节点统计量 **Traffic Received**（**bits/sec**）。

7）为网络中的所有节点，设置被提升属性 **Interarrival Time**（**seconds**）的值。回顾一下，需要使用通配符实施这项任务（见 3.5 节）。

① 以通配符替换任何被提升属性中的节点数，从而使您可在网络的所有节点中同时指定这个属性的值。

② 以通配符设置被提升属性为如下值：

- exponential（0.25）
- exponential（0.1）
- exponential（0.05）
- exponential（0.025）
- exponential（0.02）
- exponential（0.01）
- exponential（0.005）

8）运行仿真 20s。

问题

为简化所收集结果的解释，对于每个问题，可能希望使用平均函数以覆盖（overlaid）形式显示统计结果（见 4.4 节）。

Q1. 为在网络中发送的流量总量，详细研究并显示全局统计量。报文到达间隔时间的哪些值得到发送的最高流量和最低流量？为什么？回顾一下，OPNET 是依据被提升属性值的顺序，排列 DES 运行号的。因此，如果以在步骤 7 中描述的相同顺序指定被提升属性 **Interarrival Time**（**seconds**）的值，那么 DES - 1 运行将对应于设置为 exponential（0.25）的报文到达间隔时间，DES - 2 对应于 exponential（0.1）等等。

Q2. 对于网络中的任意节点，详细研究和显示类 Ethcoax 中的节点统计量 **Collision Count**。哪些仿真运行得到最高的冲突次数和次高的冲突次数？为什么？您认为这些结果与在网络中所接收的流量报告的数据一致吗？为什么一致或为什么不一致？

Q3. 对于网络中的任意节点，详细研究并显示类 **Ethcoax** 中的节点统计量 **Transmission Attempts**。哪些仿真运行得到最高的传输尝试次数和次高的传输尝试次数？为什么？您认为这些结果与在网络中所接收的流量报告的数据一致吗？为什么一致或为什么不一致？

Q4. 创建一幅图形，其中显示发送的数据量和接收的数据量。为创建这样一幅图形，实施如下步骤：

1）单击 *DES Parametric Studies* 选项卡，如图 L13.3 所示。

2）展开 **Global Statistics... Traffic Sink... Traffic Received**（**bits/sec**）统计结果。选中 sample mean 类并单击按钮 **Set As Y-Series**。

3）展开 **Global Statistics... Traffic Source... Traffic Sent**（**bits/sec**）统计结果。选中 sample mean 类并单击按钮 **Set As X-Series**。

4）单击 **Show** 按钮显示图形。

得到的图像看起来应该类似于如图 L13.3 所示的情形。解释图中所示的以太网行为。

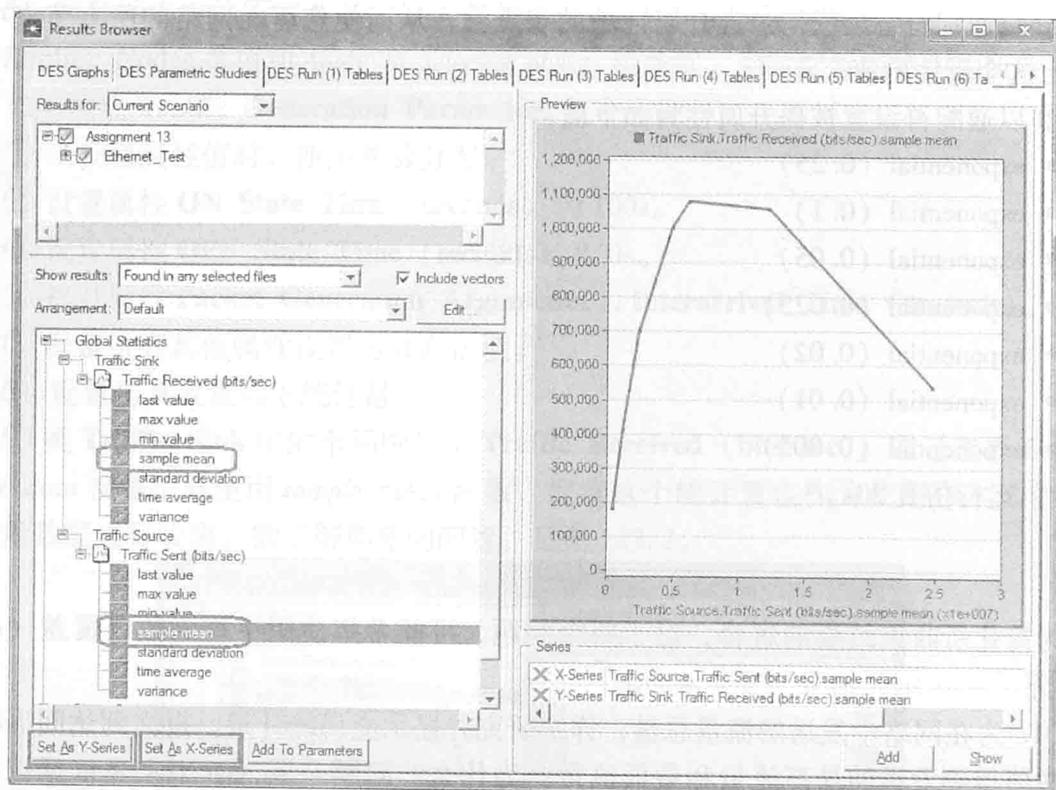


图 L13.3 使用参数化研究，分析所收集的结果

L13.3 星形拓扑

在这部分作业中，将详细研究在激活和禁止混杂模式的星形拓扑中以太网的表现。首先，将创建这样一个仿真场景，其中对禁止混杂模式的以太网络进行建模。

1）创建一个新的空场景，命名为 Ethernet_ Star。当使用 **Startup Wizard** 创建一个新场景时，使用 Office 网络规模和 ethernet_ advanced 模型族。

2）依据如下列出的规格，使用 OPNET **Rapid Configuration** 功能特征，创建如图 L13.4 所示的网络拓扑（见 2.5 节）：

① **Center node model**：ethernet32_ hub_ adv。

② **Periphery node model**：ethernet_ station_ adv。

③ **Number**：**20**。

④ **Link model**：100BaseT_ adv。

⑤ 保持其他属性设置为其默认值。

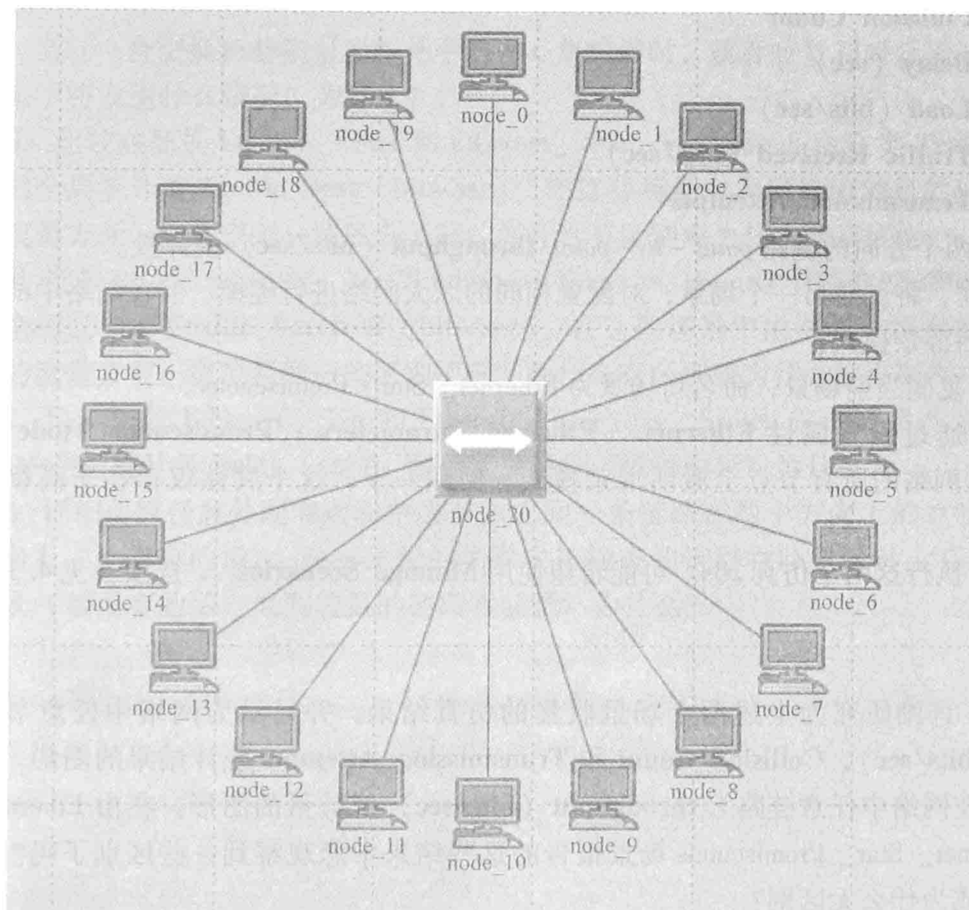


图 L13.4 以太网星形拓扑

3）验证链路连通性（见 2.6.2 节）。

4）依据如下规格，配置网络中的所有节点产生流量：

如下配置 **Traffic Generation Parameters**：

① 设置属性 **ON State Time**（**seconds**）为 constant（100）。

② 设置属性 **OFF State Time**（**seconds**）为 constant（0）。

③ 设置属性 **Packet Generation Arguments... Interarrival Time**（**seconds**）为 exponential（0.01）。

④ 保持所有其他属性设置为其默认值。

⑤ 默认情况下，在以太网节点中是禁止混杂模式的，所以为配置这项功能特征，不需要做出改变。

5）配置仿真收集如下统计量：

① 类 **Traffic Sink** 中的全局统计量 **Traffic Received**（**bits/sec**）。

② 类 **Traffic Source** 中的全局统计量 **Traffic Sent**（**bits/sec**）。

③ 类 **Ethernet** 中的全局统计量 **Delay**（**sec**）。

④ 类 **Ethernet** 中的如下节点统计量：

- **Collision Count**
- **Delay**（**sec**）
- **Load**（**bits/sec**）
- **Traffic Received**（**bits/sec**）
- **Transmission Attempts**

⑤ 两个方向的链路 *point - to - point* **throughput**（**bits/sec**）统计量。

现在，将创建另一个场景，对配置相同的以太网络进行建模，但在网络中的所有节点中激活混杂模式：

6）复制当前场景，命名新场景为 Ethernet_ Star_ Promiscuous。

7）通过设置属性 **Ethernet... Ethernet Parameters... Promiscuous Mode** 为 Enabled，在网络的所有节点上激活混杂模式。回顾一下，这个属性仅存在于高级节点模型中。

8）执行这两个仿真 20s。可能希望使用 **Manage Scenarios...** 选项（见 4. 3. 5 节）。

问题

Q5. 详细研究为上述两个场景收集的仿真结果，并显示为网络中任意节点比较 **Load**（**bits/sec**）、**Collision Count** 和 **Transmission Attempts** 统计结果的图形。同样显示比较在网络中任意链路上 **throughput**（**bits/sec**）统计量的图形。在由 Ethernet_ Star 和 Ethernet_ Star_ Promiscuous 场景报告的这些结果中您观察到一些区别了吗？为什么有区别或为什么无区别？

Q6. 详细研究为上述两个场景收集的仿真结果，并显示比较网络中任意节点的 **Traffic Received**（**bits/sec**）统计量的一个图形。您观察到在这两个场景中所收集结果之间的任何区别了吗？为什么有区别或为什么无区别？

Q7. 详细研究为这两个场景收集的全局统计量。解释观察到的行为。

Q8. 详细研究为上述两个场景收集的仿真结果，并显示比较网络中任意节点的 **Delay**（**sec**）统计量的一个图形。您观察到在这两个场景中所收集结果之间的任何区别了吗？为什么有区别或为什么无区别？

L13. 4 集线器和交换机

在这部分作业中，将详细研究以太网性能如何受一个星形网络拓扑中中心节点类型的影响。

1）复制 Ethernet_ Star 场景，命名新场景为 Ethernet_ Star_ vs_ Switch。

2）将网络中中心节点的属性 **model** 改变为 ethernet128_ switch_ adv。回顾一下，

属性 **model** 是一个高级属性，需要为之选中检查框 *Advanced*，使之变得可见。

3）为这个新场景运行仿真，并详细研究所收集的统计量。

问题

Q9. 当以一台交换机替换星形拓扑中的中心集线器时，就吞吐量和时延而言，您预期在网络中将发生什么情况？为什么？

Q10. 比较在场景 Ethernet_ Star 和 Ethernet_ Star_ vs_ Switch 中在类 **Traffic Sink** 下收集的全局统计量 **Traffic Sent**（**bits/sec**）。在这些场景中由网络中的目的地节点接收到的流量方面，您观察到一些区别了吗？为什么有区别或为什么没有区别？

Q11. 比较在场景 Ethernet_ Star 和 Ethernet_ Star_ vs_ Switch 中在类 **Traffic Sink** 下收集的全局统计量 **Traffic Received**（**bits/sec**）。在这些场景中由网络中的目的地节点接收到的流量方面，您观察到一些区别了吗？为什么有区别或为什么没有区别？

Q12. 比较场景 Ethernet_ Star 和 Ethernet_ Star_ vs_ Switch 中的流量时延，即类 Ethernet 中全局统计量 **Delay**（**sec**）。哪个场景得到较低的时延？为什么？

Q13. 详细研究任意外围节点和中心节点之间一条链路在两个方向上的吞吐量。在哪个方向上（如果有的话），在所比较的场景中被检查的链路吞吐量统计上存在不同？哪个场景（如果有的话）导致较低的链路吞吐量？为什么？

实验室作业 14：无 线 通 信

针对这项实验室作业，建议阅读第 1 ~4 章和第 12 章。

L14.1 引言

在这项实验室作业中，将探究无线通信网络的性能。特别地，在这项实验室作业的第一部分，将实施一项试验研究，以便确定采用默认 WLAN 设置进行配置的两个移动节点之间的最大通信范围。接下来，将详细研究发送功率如何影响通信范围。最后，在这项作业的最后部分，将比较 AODV 和 DSR MANET 路由协议的性能。

注意，为完成这项作业，需要有无线模块的许可证。联系 OPNET 支持，得到教育或学术研究的附加模块许可证。

L14.2 确定通信范围

首先，将建立和配置一个简单的两节点网络拓扑，其中源节点连续向目的地发送流量，而目的地节点移动远离源。在仿真过程中的某个时间点，目的地移动远离源的距离太远以致任何流量都不能通过了。基于通信停止的时间和目的地节点移动远离的速度，将确定这两个节点之间的最大通信距离。

1）创建一个新项目和一个空场景，分别命名为 Assignment 14 和 Communication Range（见 1.6.2 节）。当采用 **Startup Wizard** 创建一个新场景时，使用 Campus 网络规模并包括 wireless_ lan、wireless_ lan_ adv 和 MANET 模型族。

2）创建如图 L14.1 所示的网络拓扑：

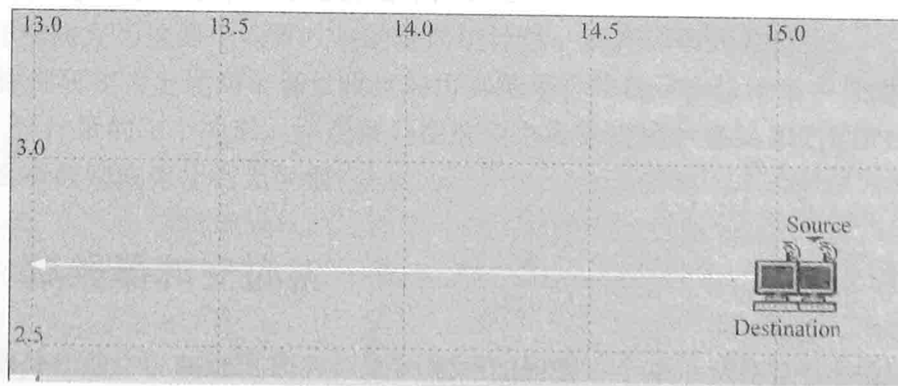


图 L14.1 简单的两节点拓扑

① 在工作空间中放置两个 wlan_ station_ adv 移动节点，并将它们命名为 source 和

destination。

② 指定节点的位置，从而使它们在 x 轴上相隔 0.1km（如 12.7 和 12.8），而 y 坐标值是相同的（如 2.8 节）。通过高级属性 **x position** 和 **y position**（即必需首先选择 *Advanced* 检查框），可指定节点 x 和 y 坐标。

3）如下定义一条新节点轨迹（见 12.8.1 节）：

① 命名新轨迹为 Distance，并配置它为相对于节点位置进行定义的（即选择标题为 *Coordinates are relative to node's position* 的检查框）。

② 在一条直线上设定轨迹长度大约为 2000m，该直线平行于两个移动节点之间的线。

③ 设置节点的速度为 2m/s。

4）配置 Destination 节点：

① 设置节点的 MAC 地址（属性 **Wireless LAN... Wireless LAN MAC Address**）为 123456。

② 设置节点的 **trajectory** 为 Distance。

5）配置 source 节点：

① 设置 **Destination Address** 为 123456。

② 如下配置 **Traffic Generation Parameters**：

- 设置属性 **Start Time**（**seconds**）为 constant（5）。
- 设置属性 **ON State Time**（**seconds**）为 constant（2000）。
- 设置属性 **OFF State Time**（**seconds**）为 constant（0）。
- 保持所有其他属性设置为其默认值。

6）配置仿真收集类 Wireless LAN 下的所有节点统计量。

7）运行仿真 18min。

问题

Q1. 详细研究并显示目的地节点接收的流量总量和源节点发送的流量总量的统计结果。在仿真过程中哪个时间点处目的地节点停止接收任何数据？基于这个值，计算这两个节点之间的最大通信距离。回顾一下，配置目的地节点以 2m/s 的速度运动，且最初时各节点是相隔 100m 放置的。

Q2. 正如在 Q1 所显示的图形中观察到的，在节点运动得太远以致任何数据都不能成功地到达目的地之后，源节点的发送速率增加。就为什么会发生这种情况，写下您的假设。详细研究并显示在源节点收集的如下统计量：**Backoff Slots**（**slots**）、**Retransmission Attempts**（**packet**）、**Control Traffic Rcvd**（**bits/sec**）和 **Queue Size**（**packets**）。同样详细研究并在单个覆盖式图形中显示 **Data Traffic Sent**（**bits/sec**）和 **Load**（**bits/sec**）。基于这些统计量给出的结果，解释所发生的情况。这些结果支持您最初的假设吗？为什么支持或为什么不支持？

L14.3 通信范围和发送功率

在这部分作业中，将详细研究发送功率如何影响通信范围。

1）复制当前场景，命名新场景为 Communication Range vs Power。

2）提升在源节点中的 **Transmit Power**（W）属性。

3）将被提升的属性设置为如下值：0.001、0.002、0.005 和 0.01。

4）执行仿真 18min。

问题

Q3. 详细研究仿真结果，并为发送功率的每个值确定通信范围。画一幅图形，显示通信范围与发送功率的关系。

L14.4 MANET 通信：DSR 和 AODV

在这部分作业中，将比较 AODV 和 DSR 路由协议的性能。

1）创建一个新场景，命名为 AODV，使用 Campus 网络规模并包括 MANET 模型族。

2）使用无线部署导引（Wireless Deployment Wizard）（见 12.9 节），创建如图 L14.2 所示的 MANET 网络拓扑。当部署无线网络时，除非另外指明，否则使用默认设置：

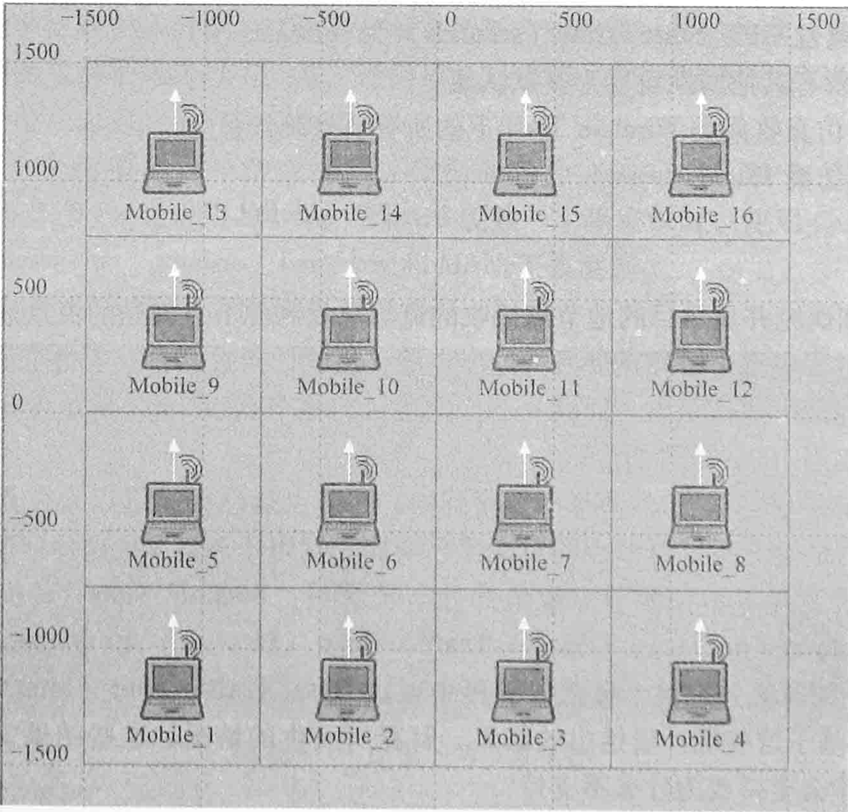


图 L14.2 MANET 格形网络拓扑

① 如图 L14.3 所示设置无线技术为 WLAN（Ad－hoc）。

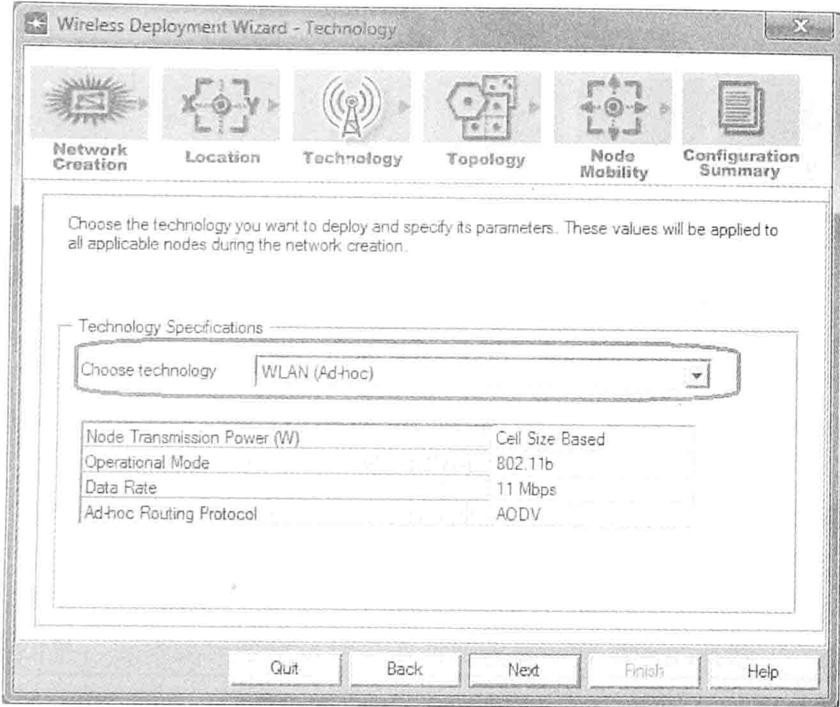


图 L14.3　在无线部署导引中的 **Technology** 配置屏幕

② 如图 L14.4 所示，设置部署区面积为 9 000 000m^2 并使用 4×4 网格节点放置。

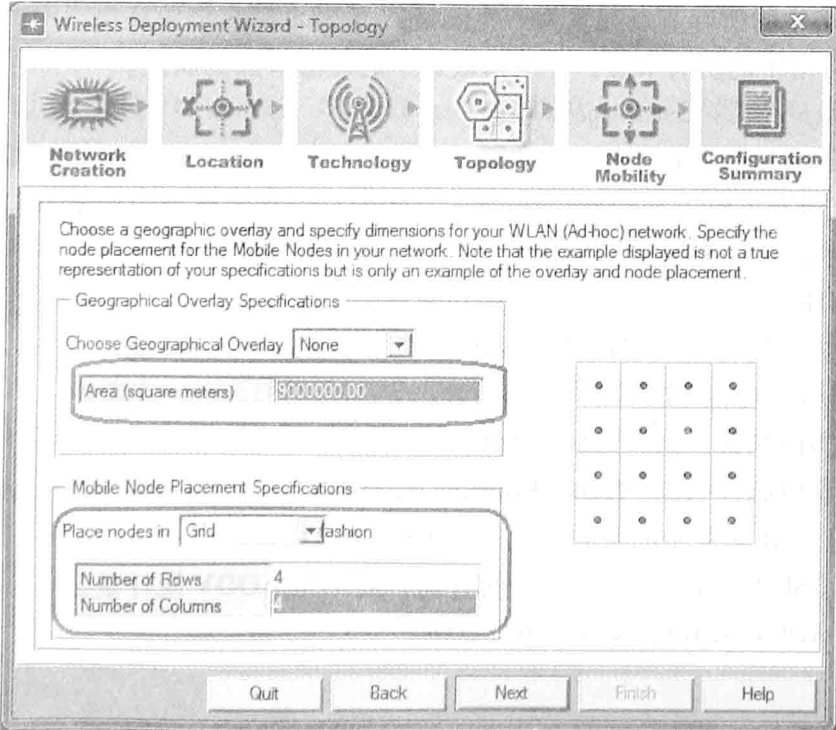


图 L14.4　在无线部署导引中的 **Topology** 配置屏幕

③ 为节点移动性使用默认设置，如图 L14.5 所示（应该不需要配置改变）。

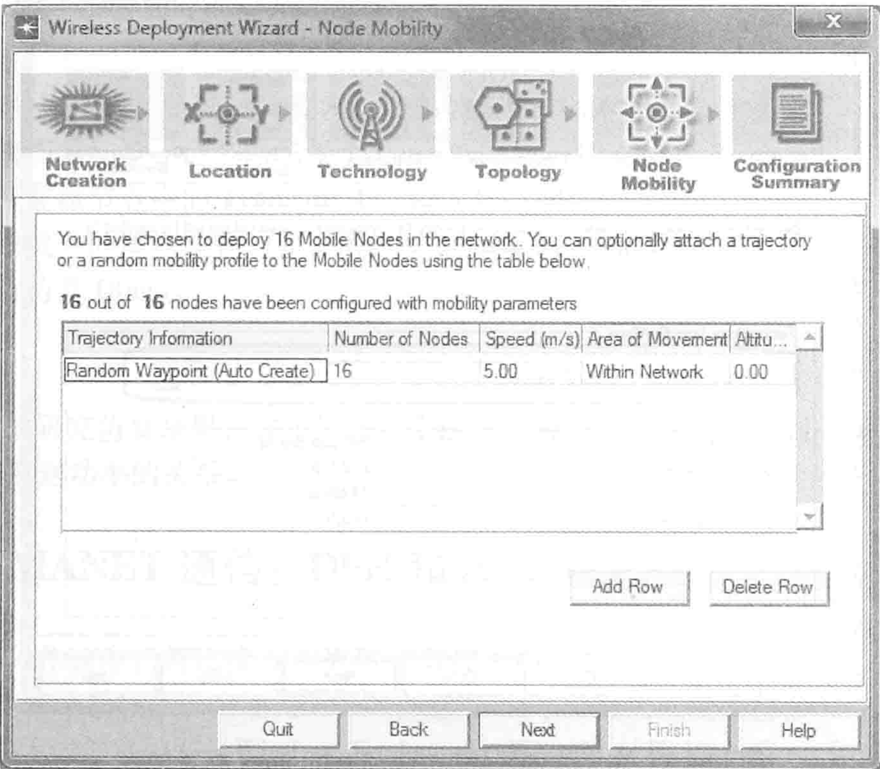


图 L14.5　在无线部署导引中的 **Node Mobility** 配置屏幕

3）将网络中的所有节点配置为产生流量：

① 选择网络中的所有节点，并编辑被选中的任意节点的属性。

② 激活 *Apply to selected objects* 检查框。

③ 设置属性 **MANET Traffic Generation Parameters... Number of Rows** 为 1，这会激活默认流量产生模式（即从时间 100s 时开始与一个随机选中的节点通信，每隔 1s 产生尺寸为 1024 字节的报文）。

④ 单击 **OK** 按钮，将这些改变应用到网络中的所有节点。

4）配置仿真所有全局 **AODV**、**MANET** 和 **Wireless LAN** 统计量（AODV 是一个 MANET 中的默认路由协议）。

5）复制当前场景，命名新场景为 DSR。

6）通过设置属性 **AD－HOC Routing Parameters... AD－HOC Routing Protocol** 为 DSR，配置 DSR 场景中的所有节点运行 DSR 路由协议。

7）配置 DSR 场景收集全局 **DSR** 统计量，而不是 **AODV** 统计量。

8）执行 AODV 和 DSR 这两个场景 200s。

问题

Q4. 给出 AODV 和 DSR MANET 路由协议的一个简短描述，突出它们的区别和相

似性。

Q5. 比较并显示 AODV 和 DSR 场景的全局统计量，如 **Number of Hops per Route**、**Route Discovery Time** 和 **Routing Traffic Received**（**bits/sec**）。所报告的结果与如下 MANET 环境中 AODV 和 DSR 的期望行为一致吗？这种环境中所有节点都移动并与网络中任何其他随机被选中的节点通信。为什么一致或为什么不一致？

Q6. 比较并显示 AODV 和 DSR 场景的全局 MANET 统计量，如 **Delay**（**sec**）、**Traffic Received**（**bits/sec**）和 **Traffic Sent**（**bits/sec**）。所报告的结果与 AODV 和 DSR 路由协议的期望行为一致吗？为什么一致或为什么不一致？

Q7. 比较并显示 AODV 和 DSR 场景的全局无线局域网统计量，如 **Data Dropped**（**Retry Threshold Exceeded**）（**bits/sec**）、**Delay**（**sec**）、**Load**（**bits/sec**）、**Retransmission Attempts**（**packets**）和 **Throughput**（**bits/sec**）。所报告的结果与 AODV 和 DSR 路由协议的期望行为一致吗？为什么一致或为什么不一致？

Q8. 如果再次运行仿真但带有一个不同的种子值，您认为所收集的结果将展示稍微不同的 AODV 和 DSR 行为吗？为什么有不同或为什么没有不同？以 5 个不同的种子值重新运行 AODV 和 DSR 场景（可通过 **Configure/Run DES** 对话框指定多个种子值）。使用所收集的结果，支持或反驳您对种子值对所收集结果影响的假定。

Q9. 比较仅有两个通信节点的 MANET 网络中 AODV 和 DSR 行为：

1）复制 AODV 和 DSR 场景，命名新的场景分别为 AODV_ one_ comm_ pair 和 DSR_ one_ comm_ pair。

2）如下修改每个新场景：

① 在网络中的所有节点上禁止流量产生。

② 设置节点 Mobile_ 16 的 IP 地址为 192. 0. 1. 1。

③ 通过设置属性 Destination IP address 为 192. 0. 1. 1，配置节点 Mobile_ 1 向 Mobile_ 16 发送流量。

3）运行新的场景，并对仅有一个通信节点的场景再次回答问题 Q5 ~ Q8。

Q10. 比较如下 MANET 网络中 AODV 和 DSR 行为，其中在整个仿真中各节点保持静态不动：

1）复制 AODV 和 DSR 场景，命名新的场景分别为 AODV_ static 和 DSR_ static。

2）修改两个新场景，从而使网络中的所有节点的轨迹均设置为 NONE（即各节点将不会到处移动）。

3）运行新场景，并针对在仿真过程中各节点不会移动的场景再次回答问题Q5 ~ Q8。

国际视野　科技前沿

国际信息工程先进技术译丛

《计算机网络仿真OPNET实用指南》
《移动无线信道》（原书第2版）
LTE-Advanced：面向IMT-Advanced的3GPP解决方案
声学成像技术及工程应用
认知无线电通信与组网：原理与应用
LTE/SAE网络部署实用指南
网络性能分析原理与应用
云连接与嵌入式传感系统
IP地址管理原理与实践
自组织网络：GSM，UMTS和LTE的自规划、自优化和自愈合
实现吉比特传输的60GHz无线通信技术
LTE自组织网络（SON）：高效的网络管理自动化
UMTS中的LTE：向LTE-Advanced演进（原书第2版）
无线传感器及执行器网络
UMTS中的WCDMA－HSPA演进及LTE（原书第5版）
认知无线电网络
网络融合 ——服务、应用、传输和运营支撑
UMTS中的LTE:基于OFDMA和SC-FDMA的无线接入
高性能微处理器电路设计
吉规模集成电路互连工艺及设计
高级电子封装（原书第2版）
基于4G系统的移动服务技术
移动无线传感器网 ——技术、应用和发展方向
UMTS蜂窝系统的Q0S与Q0E管理
UMTS-HSDPA系统的TCP性能
基于射频工程的UMTS空中接口设计与网络运行
未来UMTS的体系结构与业务平台：全IP的3GCDMA网络
环境网络：支持下一代无线业务的多域协同网络
基于蜂窝系统的IMS—融合电信领域的V0IP演进
蜂窝网络高级规划与优化 2G/2.5G/3G/ ——向4G的演进
微电子技术原理、设计与应用
多电压CMOS电路设计
P2P系统及其应用
IPTV与网络视频：拓展广播电视的应用范围
下一代无线系统与网络







上架指导　计算机／计算机网络

ISBN 978-7-111-47060-1　定价：99.00元

[General Information]
书名=计算机网络仿真OPNET实用指南
页数=419
SS号=13610637